SIERPIEŃ 1986. CENA 100 ZŁ
ISSN 0860-25-14

KOMPUTER

5

POPULARNY MIESIĘCZNIK INFORMATYCZNY



KOMPUTER

NR 5

Popularny Miesięcznik Informatyczny – pismo miłośników i użytkowników mikrokomputerów redagują:

Marek Młynarski (red. nacz.)
Władysław Majewski (z-ca red. nacz.)
Grzegorz Eider (sekr. red.)
Elżbieta Bobrowska (z-ca sekr. red.)
Grzegorz Czapkiewicz (programy)
Stanisław Królak (dz. zagraniczny)
Jerzy Pusiak (kasety)
Zenon Rudak (sprzęt)
Darosław J. Toruń (gry)
Tomasz Zieliński (listy)
Krzysztof Krupa
oraz współpracownicy:
Andrzej Bączyński (Łódź), Rafał Brzeski, Marek Car, Andrzej Kadlof, Jarosław Kania, Agnieszka i Zbigniew Kasprzyccy, Krzysztof Kuryłowicz (Łódź), Jacek A. Likowski, Andrzej J. Piotrowski, Juliusz Rawicz, Leszek Rudak, Grzegorz Szewczyk, Jakub Tatarkiewicz, Piotr Norbert Tymochowicz, Roland Wacławek (Katowice), Tadeusz Wilczek, Andrzej Załuski (Kraków)

Redakcja graficzno-techniczna:
Stefan Szczypka (kier.)
Małgorzata Luzińska
Beata Maruszewska

korekta: Maria Omiecińska, Romualda Miarecka

Wydawca: Krajowe Wydawnictwo Czasopism RSW "Prasa-Książka-Ruch", ul. Noakowskiego 14, 00-666 Warszawa, tel. centr. 25-72-91 do 93
Redakcja: ul. Mokotowska 48, 00-543 Warszawa, tel. 21-76-58 telex 815664 cestud pl (gości nas Warszawskie Centrum Studenckiego Ruchu Naukowego ZSP)
Skład i druk: Prasowe Zakłady Graficzne, Łódź, ul. Armii Czerwonej 28. Zam. 1940/86
Cena: 100 zł P-78

Prenumerata: kwartalnie – 300 zł, półrocznie – 600 zł, rocznie – 1200 zł. Prenumeratę od instytucji przyjmują oddziały RSW, a od osób prywatnych poczta (na wsi także doręczyciele). Prenumeratę ze zleceniem wysyłki za granicę (droższą o 50% dla osób prywatnych i o 100% dla instytucji) przyjmuje Centrala Kolportażu RSW, ul. Towarowa 28, 00-958 Warszawa, NBP XV O/M W-wa 1153-201045-139-11.
Prenumerata przyjmowana jest na II, III i IV kwartał oraz na II półrocze z miesięcznym wyprzedzeniem, a na rok następny do 10 listopada.

Ogłoszenia przyjmuje Biuro Reklamy, ul. Mokotowska 5, tel. 25-35-36; adres dla korespondencji w sprawach ogłoszeń: Noakowskiego 14, 00-666 Warszawa. Zamawiając ogłoszenia listownie należy podać datę i miejsce wpłaty (konto KWCz: NBP III O/M W-Wa 1036-5294 z zaznaczeniem "ogłoszenie w KOMPUTERZE").

1 cm² ogłoszenia kosztuje 300 zł, najmniejsze ogłoszenie – 2100 zł, cała strona – 200 tys. zł; kolor dodatkowy – 30% drożej, pełna gama barw – 100% drożej. Za treść ogłoszeń redakcja nie odpowiada.

Nakład 150.000 egz.
Nr indeksu 36-345 ISSN 0860-2514

MENU

W "Komputerze" i okolicach

Wydawałoby się, że okres lata nie sprzyja bardziej intensywnym pracom, poza zbieraniem ogórków. Tymczasem dzieje się wiele, podejmowane są konkretne decyzje i jesienią, kiedy to zbiera się plony ziemiopłodów, jabłek i szeregu innych dobrych rzeczy, również i miłośnicy mikrokomputerów będą mogli zapoznać się z efektami zawartych umów i transakcji. A więc po pierwsze: komputery pojawią się w sklepach Centralnej Składnicy Harcerskiej! Ceny będą do przyjęcia i należy sądzić, że pierwsza partia sprzętu w ilości kilku lub kilkudziesięciu tysięcy sztuk będzie w sprzedaży dłużej niż jeden dzień. Są też poważne szanse na powtórzenie takiej transakcji, o ile pierwsza się uda, przysparzając CSH zysków, a jej klientom dobrego sprzętu po niewygórowanej cenie. Kolejny szturm na sklepy, w których płaci się "zielonymi" zapowiada PEWEX. Firma ta z powodzeniem odrabia swoje znaczne opóźnienie w wejściu na mikrokomputerowy rynek, jak i prostuje nieporozumienia z okresu pierwszych zakupów. W ofercie PEWEX-u dominuje Atari i ma być dostępny nawet taki przebój jak Atari 520 ST. A ceny? O pieniądzach dżentelmeni (i nieliczne zainteresowane komputerami damy) nie rozmawiają. Hasło "Atari" wywołało zresztą redakcyjny podział na dwie takie same liczebnie grupy. Rzecz w tym, że jedni w swoich marzeniach widzą się za klawiaturą Atari 520 ST lub 1040, zaś drudzy przed podobną maszynerią, ale ze znakiem firmowym IBM. Czysto platoniczne uczucia, które obu grupami miotają postaramy się opisać w następnych numerach "Komputera". Realnie bliżej jest jednak coś IBM – o podobnego, bowiem w pełni zgodna z IBM-XT "Mazovia" jest najnowszym faktem naszego, polskiego przemysłu. Do tego tematu także powrócimy.

Pełną parą od czasu do czasu – jak to w lokomobili – pogwizdując, idzie produkcja kaset z programami komputerowymi. Tytułów powoli przybywa, a przy okazji apelujemy do Czytelników o korespondencję w dwu sprawach:

1. Jakiego typu programy powinny być wydawane w naszym wydawnictwie;
2. Zgłaszanie i przesyłanie do naszego wydawnictwa ciekawych programów wszystkich typów.

Jeżeli program będzie się kwalifikował do wydawania, informuję wszystkich chętnych, że nasze stawki są nader atrakcyjne! Korespondencję w tych sprawach prosimy kierować na adres redakcji z koniecznym oznaczeniem na kopercie – redakcja programów komputerowych. Gorąco zachęcam wszystkich amatorskich i profesjonalnych programistów, tym bardziej, że poza całkiem sporą paczką pieniędzy czeka na autorów SŁAWA należna twórcom pierwszych polskich programów mikrokomputerowych. Wydawane obecnie przez nas programy opracowane są na Spectrum, ale nie tylko ten typ komputera jest w Polsce (chociaż go najwięcej) i z chęcią będziemy rozważali propozycje na inne komputery, a także na inne nośniki informacyjne tj. nie tylko kasety, ale i dyskietki (chyba jednak nieco później).

Kasety z naszymi programami komputerowymi dostępne będą na początek w klubach MKPiK w miastach: Bydgoszczy, Gdańsku, Katowicach, Krakowie, Koszalinie, Kielcach, Łodzi, Olsztynie, Poznaniu, Rzeszowie, Szczecinie, Warszawie, Wrocławiu. Ciągle nie wiadomo, jakie będzie zapotrzebowanie, obiecujemy dostosowanie się do potrzeb.

Z kolei mikrokomputery czekają na zwycięzców wspólnego konkursu "Komputera" i "MAGAZYNU RAZEM" na scenariusz i w drugiej fazie na program komputerowy upowszechniający wiedzę historyczną i przydatny do ciekawej nauki historii. Tematyka będzie dowolna, szczegóły i regulamin konkursu podamy w następnych wydaniach "Komputera" i "MAGAZYNU RAZEM", chcemy jednak już zasugerować temat: bitwa pod Grunwaldem i wszelakie związane z nią wydarzenia polityczne i militarne.

Z przyjemnością informuję, że systematycznie rozrasta się redakcyjna baza komputerowa. Sądzę, że dysponujemy obecnie prawie wszystkimi komputerami dostępnymi na polskim rynku, a nasze doświadczenia z ich użytkowaniem będziemy na bieżąco publikować. Testujemy obecnie komputer mało – jak dotychczas – znanej firmy LASER. Jest to typ Laser 700 ze stacją dysków i zielonym monitorem. Jeżeli jakość tego i innych komputerów firmy LASER dorównuje ich rewelacyjnie niskiej cenie, wówczas być może już niedługo firma ta przejdzie z pozycji nieznanego Kopciuszka na pozycję ogólnie uwielbianej królewny. Dziś jednak zbyt wcześnie jest na bardziej stanowcze stwierdzenia, chcemy bowiem ze spokojnym sumieniem polecać sprzęt nie tylko tani, ale i dobry oraz niezawodny. Test komputera umożliwił nam inż. KAJETAN PYRZYŃSKI, który, jak sam mówi, szukał po prostu dobrego i najtańszego komputera.

I wreszcie ten rodzynek, który schowany jest na koniec, niby to sprawa wewnątrzredakcyjna, ale na pewno interesująca Czytelników. Witamy w "Komputerze" Andrzeja J. Piotrowskiego! Wspominałem Andrzeja w poprzednim numerze naszego pisma, nie wiedziałem wówczas, że podejmie z nami współpracę. Z tym większą przyjemnością mogę go powitać, a od nr 6 naszego pisma zachęcam do czytania opracowywanych przez Andrzeja stron.

Wszystkie nasze działania redakcyjne podejmujemy z myślą o Czytelnikach i dlatego ważne są dla nas Wasze listy. Mamy nadzieję, że skomputeryzowany bank listów do redakcji, prowadzony przez Tomka Zielińskiego pozwoli nam na bliskie kontakty, może nawet IV stopnia. Piszcie do nas!
Wszystko czytamy, a na większość listów odpowiadamy!

MAREK MŁYNARSKI

Witamy Juniora!

Cykl produkcyjny "Komputera" staje się powoli w pełni zgodny (kompatybilny) z otoczeniem programowym, w którym funkcjonuje nasze pismo, tak więc zsyłając w ostatniej chwili komentarz do numeru sierpniowego zdążę się jeszcze podzielić z Wami majowymi wieściami.

Mówiąc o cyklu produkcyjnym "Komputera" warto powiedzieć kilka słów o tym, jak pismo mikrokomputerowe powstaje: gdzie jesteśmy Turbo, a gdzie raczej mono, szaro i piechotą. Nasi współpracownicy piszą swe artykuły w większości na maszynie do pisania, chociaż kilku z nich, i to najaktywniejszych, korzysta z własnego komputera. Najczęściej jest to Spectrum, w jednym wypadku aż MacIntosh. Przygotowany na komputerze tekst drukowany jest na prywatnej drukarce autora lub dostarczany do redakcji w postaci kasety.

W redakcji sekretarz dysponuje co prawda Spectrum, lecz stojącym w pobliskiej ciemnicy, tak więc tekst dostarczony na dyskietce woli dać pierwszemu nieszczęśnikowi z własną drukarką, który się nawinie, by polecić mu wydrukowanie.

Tak uzyskane maszynopisy autorskie są ponownie przepisywane "na czysto" – w czterech egzemplarzach. Gotowe już maszynopisy gwarancyjne zawożone są do Łodzi, gdzie w dziale fotoskładu drukarni wprowadzane są ponownie do pamięci komputera, drukowane w postaci tzw. szpalt, których korekta przeprowadzana jest sposobem tradycyjnym: jedna pani czyta, druga poprawia. Naniesione ołówkiem poprawki wprowadzane są do pamięci komputera, po czym na ekranie odpowiedniego kształtu i rozdzielczości odbywa się tzw. łamanie kolumny: rozmieszczanie tekstu na stronie, wprowadzanie prostych elementów graficznych, jak linie i znaki specjalne itp.

Gotowe strony (tzw. kolumny) naświetla się i dalej krążą one w postaci przezroczy o rozmiarach strony pisma: wracają do Warszawy do kolejnej, redakcyjnej korekty. Jeszcze chwila w Urzędzie Kontroli i można rozpoczynać montaż: naklejanie kolumn na arkusze folii o rozmiarach 8 dziwacznie na pozór złożonych ze sobą stron pisma. W trakcie montażu wkleja się zwykłą taśmą przylepną elementy graficzne oraz ilustracje.

Ilustracje mają za sobą własną drogę: od MacIntosha, Spectrum lub Atari, na którym zostały przygotowane i wydrukowane, poprzez scanner rozkładający je na tzw. wyciągi barwne do gotowych do naklejania na montowany arkusz klisz.

Jeszcze druk, półmechaniczne składanie numerów z poszczególnych arkuszy, przycinanie, szycie i kolportaż.

Omówiłem tak szeroko dzieje pisma by pokazać Wam nasz punkt wyjścia. Pisząc o pożytkach z komputerów chcemy sami dawać ich przykład. Będziemy Wam co kilka miesięcy przedstawiać nasze sukcesy i porażki na tej drodze – tak jak dotychczas systematycznie staraliśmy się śledzić losy zamówienia rządowego na komputer oświatowy. Dziś gorąco witamy ELWRO 800 Jr – szerzej przedstawimy go za miesiąc, razem z relacją z Targów Poznańskich, Infovideo 86 – tym razem bardzo udanej inicjatywy "Abakusa" i jego prezesa.

WŁADYSŁAW MAJEWSKI

Informatyka i środki masowego przekazu

Fragmenty dyskusji na forum seminarium PRESSINFO (Forum Prasy i Informacji), zorganizowanego w Walencji w 1985 r.

RICHARD H. LEONARD, przewodniczący Międzynarodowego Instytutu Prasy (IPI), wydawca i wiceprezes "The Milwaukee Journal":

Nie ma już sposobu zawrócenia fali informacji zalewającej cały nasz świat. Jest tylko kwestią sporną, kiedy ta fala nas porwie.

Moim zdaniem, środki masowego przekazu znacznie lepiej umieją posługiwać się informatyką, niż rozumieją i przedstawiają jej wpływ na stan społeczeństwa.

Współczesny wydawca znajduje się nagle w sytuacji, gdy wokół niego jest znacznie więcej znaków zapytania niż odpowiedzi. I to nawet wówczas, gdy dotyczy to jego własnej gazety. Główne, warte zastanowienia kwestie to:

- Czy w sytuacji, gdy środki masowego przekazu stają się coraz bardziej uzależnione od satelitów, mikrofal i innych środków elektronicznego przenoszenia informacji, stają się one zarazem coraz bardziej zależne od kontroli rządowej? (co jest całkiem możliwe)
- Czy bezpośrednie przekazywanie reklam do domów potencjalnych klientów ograniczy dochody gazety z ogłoszeń, będące podstawą jej egzystencji? (pewien spadek wpływów wydaje się bardzo prawdopodobny)
- Co stanie się z systemem pocztowym używanym obecnie do kolportażu gazet, jeżeli tak zwane "elektroniczne listy" (teksty przekazywane liniami telekomunikacyjnymi monitor–monitor komputera – przyp. tłum.) staną się powszechne? (z pewnością opłaty za tradycyjne przesyłki zostaną podniesione)

Osobną kwestią są efekty uboczne rozpowszechniania się szybkich systemów komunikacji elektronicznej i powiększania się pozbawionego granic świata swobodnego przepływu informacji, składającego się z producentów mikrokomputerów, właścicieli satelitów. Dla przykładu, dzieci w Meksyku uczą się szybciej liczyć po angielsku niż po hiszpańsku, ponieważ wolą oglądać amerykańskie programy telewizyjne dla najmłodszych.

Jak niewiele wie opinia publiczna, nawet krajów wysoko rozwiniętych, o tym co dzieje się w świecie informatycznego giganta!

Ludzie szybciej doświadczają efektów wpływu informatyki na życie społeczne, niż dowiadują się o ich istnieniu.

CEES HAMELINK, profesor nauk społecznych uniwersytetu w Amsterdamie:

Technologia komputerowa funkcjonuje optymalnie w sformalizowanych i scentralizowanych strukturach. Te cechy umacniają tendencje do koncentracji w świecie zbierania i rozpowszechniania informacji, pogarszając jednocześnie i tak już nierównomierny dostęp do niej oraz ułatwiający atomizację jej dystrybucji. Ogranicza to poważnie możliwość demokratycznego formowania opinii publicznej.

Środki masowego przekazu nie tylko że nie demistyfikują "ery informatycznej", ale, wręcz przeciwnie, mają skłonność do euforycznego zachłystywania się "technologicznym determinizmem"

tak wyraźnie widocznym w zachodniej szkole myślenia oraz do propagowania adaptowania się społeczeństwa do innowacji technicznych, które, jak się oczekuje, usprawnią życie codzienne. Środki masowego przekazu powinny pomagać społeczeństwu w osiąganiu stanu dojrzałości psychicznej, w którym proces podejmowania decyzji, jeśli trzeba, może być korygowany. Powinny raczej brać udział w kształtowaniu socjotechnicznej czujności niż w programowaniu socjokulturowej adaptacji.

ELEAZAR DIAZ RANGEL, prezes Latynoamerykańskiej Federacji Prasowej:

Przez cztery ostatnie stulecia technologia stosowana przez gazety, z wyjątkiem maszyny do pisania, miała tylko niewielki wpływ na zawód dziennikarza. Studium dotyczące prasy w okresie przejściowym, przygotowane przez XVI Zgromadzenie Międzynarodowego Instytutu Prasy, jakie odbyło się w Genewie w czerwcu 1967 roku, stwierdza, że

jesteśmy świadkami rewolucji technicznej, której "olbrzymia waga nie znajduje porównania w historii środków masowego przekazu".

DENIS HOLDER pisał w "Washington Journalism Review" w styczniu 1983 roku, że istnieje tyle metod produkowania gazety elektronicznej, ilu wydawców, którzy pokusili się na taki eksperyment. Tym niemniej w raporcie McBride'a stwierdzono, że chociaż "świat wchodzi w wiek elektroniki, to jednak nic nie upoważnia do przepowiadanią, że drukowane media zanikną".

W gazetach przechodzących na system komputerowy pojawiają się nowe zawody i miejsca pracy, chociaż nie jest jeszcze całkowicie jasne czy nowe czynności mają być wykonywane przez dziennikarzy, przez techników, czy też przez specjalistów nowych zawodów.

Niektóre z wprowadzanych zmian mają pozytywny wpływ na zawód dziennikarski. Przyspieszają przekazywanie informacji, ułatwiają redagowanie materiałów, archiwizowanie dokumentów oraz czynią redakcyjne archiwa łatwiej i szybciej dostępne. Nowe technologie dają też dziennikarzowi bezpośredni dostęp do banków danych, czynią skład i przygotowanie matryc swobodniejszymi, a sekretarzowi redakcji i wydawcom pozwalają na szersze kontrolowanie edycji.

Z drugiej strony jednak należy odnotować wpływ negatywny, który pojawia się, gdy – a często się to zdarza – zainteresowanie kierownictwa redakcji nowymi technologiami wynika jedynie z chęci ograniczenia kosztów produkcji, oszczędności czasu składu czy też ulepszenia procesu druku, a nie z chęci poprawienia merytorycznych zawartości gazety.

MAX ROLLAND Międzynarodowa Federacja Prasowa:

Eksplozja elektronicznej komunikacji może mieć wpływ na swobodę wyrażania myśli przez dziennikarza, może ułatwić cenzurę i ingerencję w artykuł dziennikarza oraz ograniczyć odpowiedzialność dziennikarza za słowo drogą obciążenia go obowiązkami, które nie on powinien wykonywać.

Wprowadzanie nowych technologii nie zawsze przebiegało bez incydentów. Londyński "The Times" spara-

lizowany był przez przeszło cztery miesiące. Opór notowano też w redakcjach i drukarniach "The New York Times" i "The Washington Post". W wielu krajach europejskich odbyły się długie negocjacje między kierownictwami gazet a związkami zawodowymi reprezentującymi dziennikarzy i drukarzy.

W swoim dążeniu do redukowania kosztów produkcji wielu wydawców z krajów rozwijających się wprowadza zaawansowane technologie bez oglądania się na stan technologicznego rozwoju własnego kraju, na wpływ nowych technologii na życie społeczne lub na jakość przekazywanych informacji czy jakość pracy dziennikarza.

Podczas seminarium na temat prasy i technologii, zorganizowanego w Naiguata w Wenezueli, w listopadzie 1981 roku, wspólnie przez Latynoamerykańską Federację Prasową, Międzynarodową Federację Prasową i Krajowe Zrzeszenie Prasy Wenezueli, z udziałem delegatów z siedmiu krajów, podkreślono stanowisko dziennikarzy latynoamerykańskich: "stwierdzamy, że nowe technologie komunikacji mogą być narzędziem rozwoju, awansu społecznego i postępu w naszych krajach pod warunkiem, że zostaną użyte w sposób ostrożny, z poszanowaniem dążeń narodowych interesów społecznych. Powtarzamy ostrzeżenie, że

ten kto kontroluje technologię danego kraju, kontroluje również jego rozwój.

Jest prawem narodu wprowadzenie i przestrzeganie odpowiednich norm celem zapobieżenia niekontrolowanemu transferowi technologii z krajów uprzemysłowionych. Poszczególne kraje winny popierać niezależne, własne tworzenie nowych technologii, mając na uwadze wymagania suwerenności narodowej. Nie zgadzamy się z tymi, którzy kierując się wyłącznie chęcią obniżenia kosztów produkcji, wprowadzają nowe technologie bez oglądania się na potrzeby narodowego rozwoju oraz nie konsultując się z pracownikami zatrudnionymi przy wydawaniu prasy".

* * *

Podsumowanie przedstawionej dyskusji można sprowadzić do trzech punktów:

- Informatyka służy rozwojowi prasy i innych środków masowego komunikowania się w krajach rozwijających się, ale nowa technologia musi być wprowadzana z poszanowaniem wymagań kultury, gospodarki oraz potrzeb społecznych danego kraju, a nie tylko dla korzyści ekonomicznych. Potrzeby społeczne oraz wymagania narodowego rozwoju muszą również być brane pod uwagę.
- Wprowadzenie informatyki w prasie i innych środkach masowego przekazu winno być przenegocjowane uprzednio przez kierownictwa oraz organizacje twórcze i związkowe zainteresowanych pracowników.
- Szkoły dziennikarskie o poziomie uniwersyteckim powinny korzystać z informatyki, ale jednocześnie winny studentom zwracać uwagę na całą złożoność związanych z tym problemów, tak, by przyszli dziennikarze nie stali się bezmyślnymi narzędziami podporządkowanymi informatyce.

Tłum. z kwartalnika "Agora" nr 10, 1985. Tłumaczył i opracował R. B.

[2] Kino i komputery

W pierwszej części artykułu ("Komputer" nr 3) autor rozważał czy "wojny gwiezdne" są w ogóle możliwe. Z omówionych warunków ograniczających, takich jak na przykład "grzech pierworodny" oprogramowania (tkwiący w przyjęciu błędnej doktryny), wynika, że nie jest to realne. Istniejący już – porównywalny co do wielkości z przyszłymi "wojnami gwiezdnymi" – system komputerowy sił lotniczych USA (WWMCCS), jest w najwyższym stopniu zawodny.

Czas na dyskusję o możliwościach oprogramowania.

DLACZEGO PROGRAMY SĄ ZAWODNE?

Wynika to z samej istoty ich powstawania. Przed pojawieniem się komputerów stosunkowo niewielką liczbę stanów w jakimś urządzeniu technicznym dawało się opisać przy pomocy funkcji matematycznych. Można było zatem urządzenie dokładnie przetestować i ukryte niespodzianki (np. wibracje silnika w pewnych przedziałach obrotów) usunąć. Tam, gdzie zawodziła analiza matematyczna funkcjonowania, błędy wykrywano w praktyce.

Skonstruowawszy komputery ludzie stanęli przed urządzeniami, w których liczba stanów nieciągłych może być bardzo duża. Z natury procesów produkcyjnych wynikało, że budowano te urządzenia z wielkiej liczby małych, powtarzalnych elementów, dających się przetestować. Dzięki temu nie trzeba równie wyczerpująco kontrolować całości. Dlatego także błędy konstrukcyjne spotyka się zwykle w tych częściach komputerów, które nie są zbudowane z powtarzalnych elementów.

Zgoła inaczej jest z oprogramowaniem – rzadko występuje dostateczne uzasadnienie dla konstruowania programu o strukturze wysoce powtarzalnej, składającej się z identycznych części. Można, naturalnie, podzielić na moduły względnie samodzielne, które poddają się testowaniu łatwiej niż całość. Jednak sama liczba takich modułów i ich wzajemnych interakcji w działaniu oprogramowania stanowi utrudnienie czasami nie do opanowania przez ludzki umysł. Gdzieś i kiedyś wet najzdolniejszy programista zaczyna się gubić w swoim własnym dziele.

Ogromna liczba stanów i brak powtarzalności sytuacji prowadzi do tak skomplikowanych reprezentacji matematycznych, że ani ludzie, ani obecne komputery nie potrafią się z tym uporać. Jest wprawdzie postęp, ale powolny. Nie należy oczekiwać rewolucji w dziedzinie, w której funkcje matematyczne są trudniejsze do ogarnięcia niż sam program, który mają opisywać! Tu znowu przejawia się wyższość umysłu ludzkiego...

Intelekt ludzki nie pracuje sekwencyjnie – jak komputer – ale może dokonywać przeskoków, skrótów myślowych, pomijania zbędnych szczegółów, może przewidzieć to, co dla komputera jest nieprzewidywalne. Oprogramowanie jest natomiast tworzone przez ludzi, którzy obok swoich wewnętrznych ograniczeń, stają przed ograniczeniami komputerów i samych programów do nich.

SZTUCZNA INTELIGENCJA

Mówi się o "sztucznej inteligencji"...– uściślijmy to pojęcie. Są dwie jego definicje. Jedna – węższa – stwierdza, że "sztuczna inteligencja" to zastosowanie komputerów do rozwiązywania problemów, z którymi poprzednio musieli borykać się ludzie. Z tego punktu widzenia skomplikowane obliczenia astronomiczne, czy rachunek prawdopodobieństwa w firmach ubezpieczeniowych, wykonane teraz przez komputery – są przejawem sztucznej inteligencji. Druga definicja stawia jednak znacznie większe wymagania: mówi, że "sztuczna inteligencja" to specyficzny zestaw technik programistycznych, prowadzących do powstania oprogramowania "poznawczego", imitującego sposób rozumowania ludzi. W tym celu dokładnie wypytuje się ludzi (zazwyczaj ekspertów w jakiejś dziedzinie) o ich sposób rozumowania.

W tworzeniu sztucznej inteligencji pierwszego rodzaju zastosowanie znajdują zazwyczaj tradycyjne techniki programowania.

Przy tworzeniu sztucznej inteligencji drugiego rodzaju programista napotyka na niemożność zdefiniowania ludzkiego myślenia, brak jest bowiem danych a te, którymi dysponujemy, są niespójne i niepełne. Programy, które umożliwiają komputerowi poznawanie rzeczywistości, powstają najczęściej metodą prób i błędów dodawania nowej zasady postępowania, gdy zestaw dawnych reguł nie wystarcza. Powstaje w rezultacie oprogramowanie, którego efekty działania trudno przewidzieć, a nawet zrozumieć. Uczeni parający się badaniami nad drugim typem "sztucznej inteligencji" nie widzą nic niewłaściwego w takim podejściu, choć z punktu widzenia wojskowego, w tym "wojen gwiezdnych", niemożność testowania programu jest oczywiście wadą dyskwalifikującą. Badania te znajdują się w sferze militarnej (a dodajmy, że tylko wojsko ma w USA dostatecznie dużo pieniędzy na te niezwykle kosztowne prace) na etapach zupełnie początkowych. Bezzałogowy czołg porusza się po polu walki z prędkością do 6 km/h, czyli tak, jak idący żołnierz, a ma przy tym kłopoty z odróżnieniem cienia na drodze leśnej, który mu nie przeszkodzi, od pnia, który musi ominąć... To nie kwestia czujników, ale interpretacji przesłanych przez nie danych.

Zdaniem pewnego znakomitego znawcy sztuki programowania, sztuczna inteligencja tak się ma do inteligencji człowieczej, jak kwiaty sztuczne do prawdziwych. Z daleka może zewnętrzne podobieństwo kogoś omamić, ale już z bliska nikt nie da się nabrać.

➧ 22

I KOMPUTER ZOBACZYŁ ŚWIATŁO

Od około dziesięciu lat funkcjonują na świecie światłowodowe systemy łączności, od telekomunikacji długodystansowej po okablowanie samochodów osobowych czy samolotów wojskowych. Niestety nie w pełni można w nich wykorzystywać zalety przekazywania impulsów drogą świetlną, gdyż na wejściu i wyjściu muszą zostać one przetworzone na sygnały elektryczne.

Od dawna myślano o komputerach świetlnych, a właściwie przewidywano ich powstanie na podstawie dociekań teoretycznych. Zalety były niezwykle obiecujące, bowiem światło jest około 200 razy szybsze niż elektrony. Przy takiej samej zatem architekturze komputer świetlny byłby bez porównania szybszy.

Od lat wiele ekip badawczych pracowało na świecie nad podstawowym układem zdolnym do modulowania strumienia światła. Pierwsze liczące się osiągnięcia powstały w Wielkiej Brytanii. Jest to rozwiązanie firmy Plessey Co., oparte na arsenku galu. Zasada działania jest następująca: po podaniu napięcia stałego prędkość światła w metalizowanych częściach układu zmniejsza się, dzięki czemu na wyjściu można otrzymać promień o modulowanej intensywności, proporcjonalnej do siły sygnału napięciowego.

Również na początku tego roku zespół badaczy z Heriot-Watt University z Edynburga (Szkocja) zaprezentował na międzynarodowej konferencji w USA transfazor, czyli pierwszy na świecie świetlny obwód numeryczny. Układ opracowany pod kierunkiem prof. Desmonda Smitha zawiera trzy transfazory, które mogą mieć dwa stany – włączony i wyłączony. Można z nich budować układy kaskadowe. Szkoci zamierzają zaprezentować jeszcze w tym roku transfazor na bazie selenku cynku, a w Plessey pracuje się nad niobkiem litu. Dąży się do uzyskania układu zdolnego do pracy w temperaturze pokojowej, transfazor pracuje bowiem w ciekłym azocie.

Osiągnięcia brytyjskie porównuje się do rewolucji, jaką w elektronice spowodował przed laty tranzystor. Transfazor, poza bez porównania większą szybkością pracy, odznacza się także zdolnością do równoczesnego przetwarzania różnych problemów na tych samych obwodach, bowiem promienie świetlne nie modyfikują się nawzajem.

Perspektywa komputera hiperszybkiego (przymiotnik "super" już nie wystarcza) wywołuje niezwykle żywe zainteresowanie polityków i wojskowych. Całkiem zasadnie przypuszcza się, że jaśnie oświecone hiperkomputery pomogą uporać się z wieloma problemami obliczeniowymi "wojen gwiezdnych", gdzie w grę wchodzi gigantyczna liczba danych, przetwarzanych według niespotykanie obszernego i skomplikowanego oprogramowania, a na dokładkę wymogi co do czasu są nadzwyczaj ostre.

Prezentując transfazory nauka uczyniła wielki krok w stronę rozwiązania nie tylko problemów militarnych, lecz także sztucznej inteligencji. Pojawiły się głosy, że komputery świetlne to sprawa nie tyle przyszłego wieku, co końca obecnego.

JACEK A. LIKOWSKI

SUPERSIECZKARNIA

My, ludzie postrzegamy świat jako ciągły strumień danych. Komputery zaś muszą sobie wszystko posiekać na serie impulsów zerojedynkowych. Na przykład jedna sekunda muzyki w zapisie cyfrowym składa się z 35 000 "plasterków". Im więcej tych kawałków rzeczywistości w jednostce czasu, tym dokładniej we wnętrzu komputera odzwierciedla się świat realny.

Amerykańska firma Analog Devices Inc. opracowała konwerter dwukrotnie szybszy niż jakiekolwiek urządzenie na rynku. Może on mianowicie dokonywać 40 mln konwersji na sekundę; jest to strumień danych wystarczający, aby wypełnić 100 dysków elastycznych.

Zastosowanie tego konwertera to głównie śledzenie przebiegu reakcji chemicznych i jądrowych, a także tego, co się dzieje z samochodem po najechaniu z pełną prędkością na ścianę z litego betonu. W tym ostatnim zastosowaniu wspomnianego urządzenia za 4 000 dolarów można uzyskać znaczne oszczędności, bowiem przy dotychczasowej technice nie można było z jednej katastrofy uzyskać dostatecznej liczby danych na pełne opisanie wszystkich zjawisk i trzeba było rozbijać kolejny pojazd. (JAL)

"PIĄTKI" DLA KOMPUTERÓW

(W) Na Węgrzech zorganizowano niedawno konkurs komputerów, które znajdą zastosowanie w szkołach podstawowych i średnich. Animatorem konkursu był Instytut Informatyki i Organizacji Nauki (TUSZI).

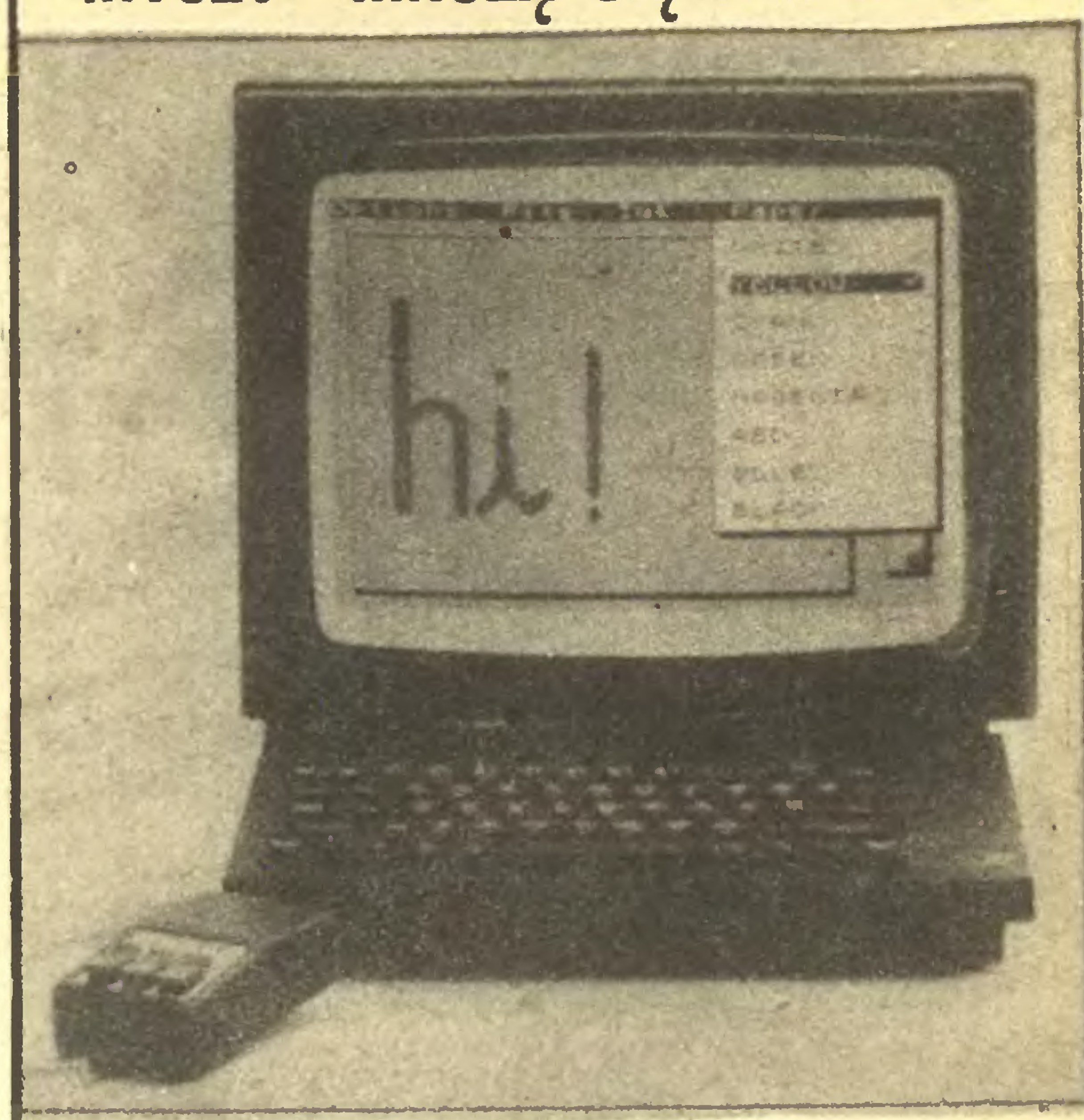
W kategorii komputerów dla szkół podstawowych pierwsze miejsce przypadło Commodore 16 (cena 7500 forintów) i komputerowi telewizyjnemu Videotonu. Ten ostatni okazał się trochę droższy (11 tys. forintów). W kategorii komputerów dla szkół średnich pierwsze miejsce uzyskał komputer produkcji węgierskiej firmy MTA-SZTAKI-Mikrokey o nazwie Pro Primo 64 K (cena 17 tys. forintów). Organizatorzy podali, że ogłoszenie wyników konkursu nastąpiło po zdeklarowaniu przez producentów obniżki cen, i to w ostatniej chwili. Videoton zgłosił 10-procentową obniżkę, natomiast producent komputera Pro Primo – 20-procentową. Nie trzeba podkreślać, że obniżka w obliczu zamówień dla całego "sektora" szkolnictwa jest w istocie doskonałym chwytem reklamowym.

W chwili obecnej w węgierskim szkolnictwie podstawowym i średnim pracuje ok. 6 tys. komputerów. Nieco ponad połowa tej liczby jest "we władaniu" szkół średnich. Plany przewidują, że „park komputerowy" szkolnictwa w drugiej połowie lat osiemdziesiątych zwiększy się o 44 tys. jednostek. W tej sytuacji w każdej większej szkole średniej będzie 8-10 komputerów.

Warto dodać, że TUSZI, czyli organizator konkursu, przesyła jego wyniki wszystkim szkołom. Nie pozostaje więc nic innego jak tylko wybierać.

Dla "Komputera" opracował:
Szczepan Woronowicz
wg Heti Vilaggazdsag

"MYSZY" MNOŻĄ SIĘ JAK MYSZY



"Myszka" do ZX Spectrum to nowa propozycja angielskiej firmy Advanced Memory Systems dla wszystkich użytkowników popularnych i u nas mikrokomputerów. Firma AMX oferuje "myszkę" z interfejsem oraz oprogramowaniem na kasecie magnetofonowej lub kasetce do microdrive'u.

Program AMX Art umożliwia posługiwanie się "myszką" – otwieranie okienek na ekranie telewizora, przeglądanie zawartości okienek, ruch wskaźnika po ekranie, wybór żądanej opcji oraz wykonywanie rysunków. Drugi program dodawany do "myszki" to "AMX Colour Palette and Control" umożliwiający wykonanie kolorowej grafiki oraz wykorzystanie "myszki" w programach użytkownika.

Wyposażony w "myszkę" mikrokomputer ZX Spectrum staje się znów atrakcyjnym narzędziem dla młodych programistów jak i ciekawą zabawką dla dzieci.

Cena tej ciekawej propozycji ("myszka", interfejs, kaseta z programami i instrukcja) wynosi 70 funtów angielskich. (ZR).

ZAMIAST MASZYNISTKI

Mikrokomputery powoli stają się narzędziem nie tylko wspomagającym człowieka, ale coraz częściej także go zastępującym. Ciągle jeszcze nie w pełni rozwiązany jest jednak problem porozumiewania się z komputerem. Coraz modniejszy ostatnio sposób komunikacji określany nazwą WIMP (Window, Icon, Mouse, Pushdown Menu) to także rozwiązanie dalekie od ideału. Najwygodniejszy w użyciu byłby oczywiście komputer reagujący na polecenia wydawane ustnie. Prace nad takimi systemami prowadzone są w wielu firmach i w najbliższym czasie przewidywane jest pojawienie się takich systemów na rynku. Niedawno amerykańska firma Dragon Systems zapowiedziała, że na początku 1987 r. rozpocznie sprzedaż oprogramowania i dodatkowej płyty do IBM PC umożliwiających przekształcenie komputera w automatyczną maszynistkę, której będzie można dyktować tekst do napisania. Podczas dyktowania trzeba będzie jednak zachowywać krótkie przerwy między poszczególnymi słowami i oczywiście posługiwać się słowami zawartymi w słowniku komputera. Słownik ten ma zawierać ponad 2000 najczęściej używanych słów.

(gs)

ZAMIAST GOSPOSI?

Inżynierowie z firmy Toshiba zapowiadają, że najpóźniej za 5 lat na rynku pojawią się roboty Toshiby przeznaczone do prac w gospodarstwie domowym. Prototyp takiego urządzenia – elektroniczna sprzątaczka mogąca odkurzyć i czyścić podłogę, ma na razie jeszcze dość duże wymiary (1,4×1,1×0,7 m), a jakość jej pracy pozostawia sporo do życzenia, zwłaszcza w miejscach trudno dostępnych. Robot porusza się w pomieszczeniach wykorzystując laser i system ultradźwiękowy

(gs)

PRZYDOMOWY ZAKŁADZIK PRODUKCYJNY KOŚCI

Oto wielkimi krokami nadchodzi nowe. Zbliża się dzień, w którym co bardziej niecierpliwy (lub świadom symboli pozycji społecznej) inżynier-elektronik będzie mógł wyprodukować sobie w domu półprzewodnikowy układzik wielkiej skali integracji. Obecnie masowa produkcja "kości" jest równie kapitałochłonna, jak hutnictwo żelaza, ale miniaturyzacja laserów i nowa technologia oparta na starej zasadzie, zwanej pantografią, sprawiają, że wizja warsztatu w garażu jest bliższa, niż mogłaby się wydawać profanom.

Nie koniec na tym: do końca 1986 r. należy oczekiwać pojawienia się prostych urządzeń do łączenia układów bramkowych (gate array) – czyli domowej technologii wytwarzania "kości" podstawowych układów. Do końca zaś lat osiemdziesiątych pojawią się na rynku biurowe (zatem całkowicie czyste!) linie technologiczne produkcji układów scalonych!

Zanim ktoś powie, że to niemożliwe, przypomnijmy, że nie tak dawno sztuka powielania słowa pisanego była praktykowana w dużych, hałaśliwych i brudnych zakładach produkcyjnych, zwanych drukarniami. Teraz zaś matryce wykonuje maszynistka lub ktokolwiek na mikrokomputerze z drukarką laserową, a rozmnożenie liczby egzemplarzy to sprawa ksero lub małego offsetu; wszystkie te urządzenia weszły na stałe do wyposażenia biurowego.

(JAL)

SZPIEG ZAWSZE CZUWA

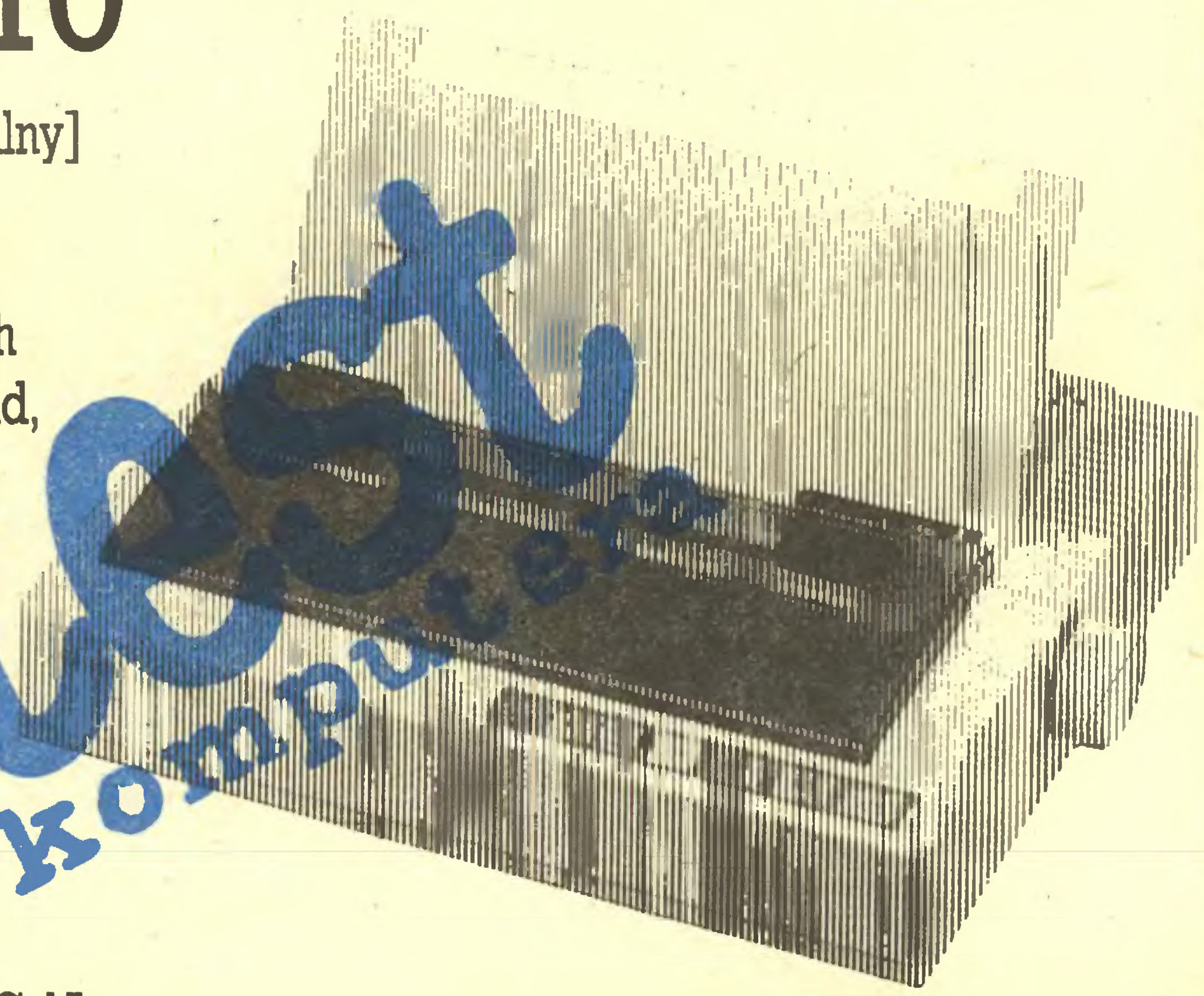
British Telecom oferuje do szyfrowania rozmów telefonicznych kość zwaną B-crypt. Będzie ona najpierw montowana do wielu własnych wyrobów koncernu, a potem oferowana jako komponent. Nadaje się do urządzeń telekomunikacyjnych i komputerów. Jej działanie polega na szyfrowaniu impulsów przesyłanych do odbiornika – którym może być np. inny komputer wyposażony w identyczny układ. Trzeba tylko uzgodnić klucz – lecz to można łatwo trzymać w tajemnicy. Natomiast dla osoby postronnej, niezależnie od sposobu przechwytywania danych, strumień informacji będzie niezrozumiały. Sposób szyfrowania jest tak skomplikowany, że odszyfrowanie wymaga ogromnych mocy obliczeniowych.

Nie jest to wprawdzie tak skuteczną ochroną tajemnicy, jak specjalne urządzenia – znane na rynku od lat – które nie dość, że szyfrują, to jeszcze stochastycznie zmieniają klucz w czasie emisji, tak że tylko identyczne urządzenie po drugiej stronie jest w stanie połapać się w niebywałym mętliku kluczy i ich przypadkowych zmian. Ale są one drogie i dlatego mało znajdowały nabywców wśród firm przemysłowych i handlowych. Ponieważ jednak okazuje się, że nie tylko kryminaliści, ale i konkurenci, potrafią czerpać wielkie korzyści z przechwytywania danych przesyłanych głosem lub cyfrowo liniami telefonicznymi, tanie i skuteczne układy do szyfrowania będą zapewne miały powodzenie.

(JAL)

NA PROŚBĘ CZYTELNIKÓW PRZEDSTAWIAMY DRUKARKI

Star NL-10



- 120 znaków/sek [tryb normalny]
- 30 znaków/sek [tryb NLQ]
- pełne możliwości graficzne
- 96 znaków programowanych
- współpracuje z IBM, Amstrad, Commodore, Atari, Spectrum, itd.
- akceptuje papier z perforacją lub bez oraz pojedyncze kartki

Drukarki marki Star, włącznie z Gemini 10X/15X i SG-15, używające łatwo dostępnej taśmy na szpulkach, oferuje ABC-Data GmbH

materiały zebrał ZR

ABC-Data GmbH
5300 Bonn 2
Post Box 200465
Telefon: 311433
Telex: 886717

Serwis, instalacja oraz informacja techniczna:
Refleks Sp. z o. o.
ul. Glogera 1
skr. pocztowa 163
02-051 Warszawa
Telefon: 6593922
Telex: 816406

GIGANT NIE ŚPI

Wszystko wskazuje na to, że w najbliższym czasie nastąpi szybkie upowszechnienie systemu Unix i mikroprocesorów typu RISC (Reduced Instruction Set Computer). Być może – znów za sprawą IBM – powstanie nawet nowy standard. Na początku tego roku IBM zaprezentowała komputer oznaczony symbolem RT PC. Zawiera on 32-bitowy mikroprocesor RISC i jest wyposażony w dysk elastyczny 1,2 M i dysk sztywny 40 M. W zależności od wersji pamięć wewnętrzna może mieć pojemność od 1 do 3 M.

W RT, podobnie jak w AT, możliwe jest rozbudowywanie systemu dzięki istnieniu 6 złączy przeznaczonych na dodatkowe pakiety. Znajdzie się wśród nich także płyta umożliwiająca wykorzystanie na RT całego oprogramowania dostępnego na rynku dla IBM PC AT.

Jako system operacyjny zastosowano wielodostępny system AIX (do 8 użytkowników). Jest to wersja zdobywającego obecnie dużą popularność systemu UNIX. RT będzie umożliwiał uzyskanie rozdzielczości 720×512 punktów lub 1024×768 punktów. RT może wykonywać od 1,2 do 2,4 mln instrukcji na sekundę. (gs)

Programy komputerowe POCZTĄ!
– dla ATARI, AMSTRAD, COMMODORE i SPECTRUM wysyła AGENCJA KOMPUTEROWA
Sosnowiec P-157

BR-189

Programy na SPECTRUM najkorzystniej otrzymasz pisząc pod adresem: SPEKTRA, 21–426 Wola Mysłowska.

BR-250

Firma MUEL oferuje do sprzedaży:
1) INTERFEJS do ZX SPECTRUM umożliwiający współpracę z czterema napędami dysków elastycznych, dowolną drukarką graficzną, monitorem ekranowym, rozszerzający BASIC oraz system operacyjny ZX SPECTRUM. Nie zajmuje pamięci RAM!
2) Sterowany "ikonami" programator EPROM 2716÷27256 do ZX SPECTRUM
3) Przeróbkę drukarki DZM 180 na drukarkę graficzną.
Informacja tel. 33-40-91
Korespondencja: MUEL
ul. Cząstkowska 30
01-678 Warszawa

BR-201

ZX SPECTRUM SERVICE, PMS ELEKTRONIK



ul. Legionowa 23, 01-343 Warszawa, skr. poczt. 17.

Poleca:

- **serwis komputerów firmy Sinclair Research Ltd.,**
- **interfejsy do joystick'ów systemu Kempston,**
- **interfejsy do drukarek systemu Centronics,**
- **rozszerzenie pamięci RAM,**
- **kable monitorowe, TV oraz inne,**
- **programy użytkowe oraz gry komputerowe dla komputera ZX Spectrum.**

BR-208

"Mikrokomputery". Pośrednictwo kupna-sprzedaży sprzętu i oprogramowania
mgr inż. Krzysztof Gewald
mgr Krzysztof Fiech
Sosnowiec, tel: 664412, w godz. 17–19"

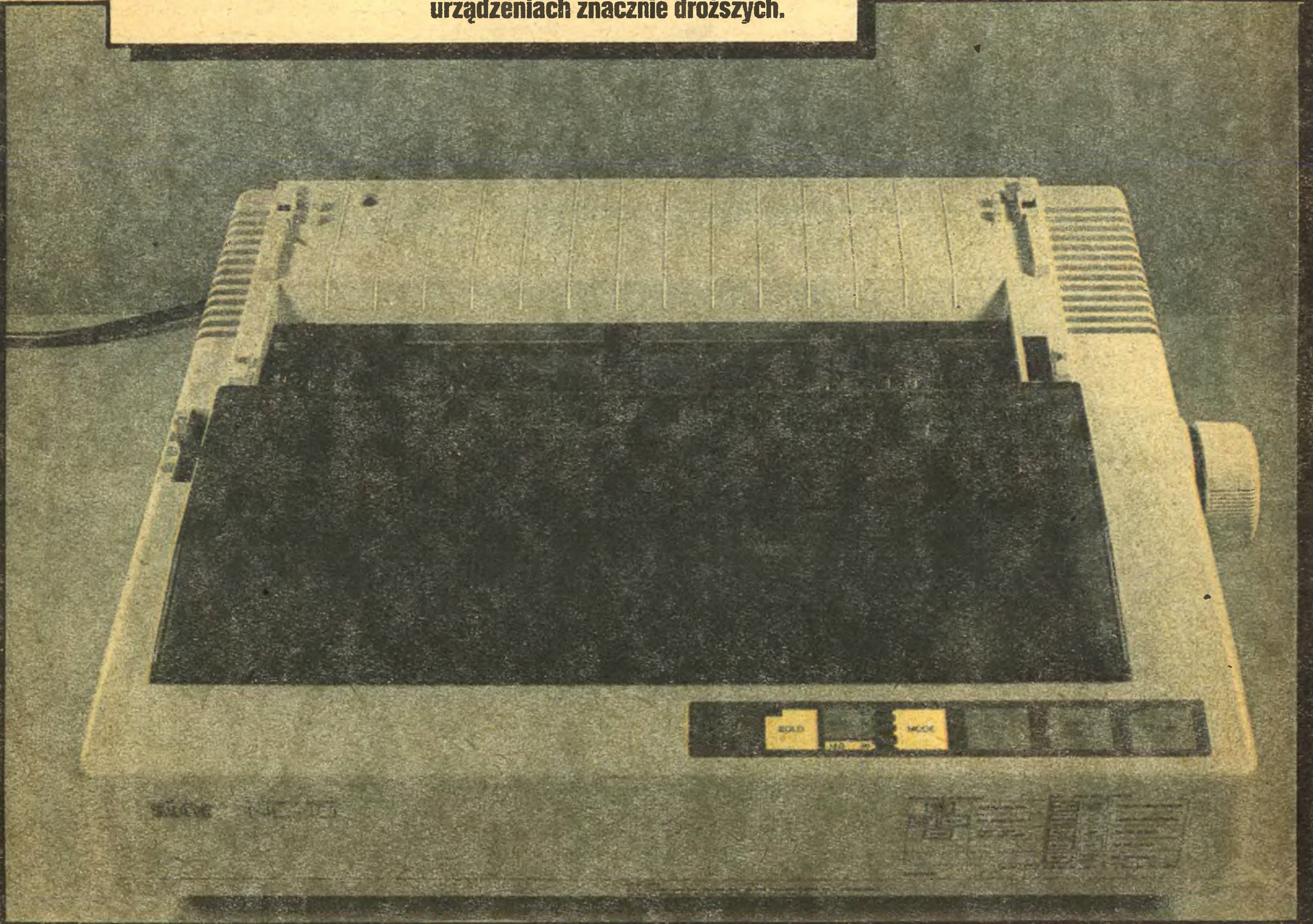
test komputera

Testem drukarki STAR NL-10 otwieramy rubrykę testów sprzętu mikrokomputerowego i urządzeń peryferyjnych. Będziemy na naszych łamach przedstawiać Czytelnikom informacje i nasze spostrzeżenia o otrzymanym do testowania w redakcji sprzęcie mikrokomputerowym.

Drukarka STAR NL-10

W nr. 3 naszego miesięcznika zamieściliśmy obszerną relację z wystawy HOME OFFICE COMPUTER, jaka odbyła się w kwietniu w Warszawie. Jednym z wystawców była firma STAR MICRONICS. Dzięki uprzejmości jej przedstawicieli redakcja mogła testować drukarkę STAR NL-10.

Drukarka STAR NL-10 posiada szerokie możliwości dostępne dotychczas w urządzeniach znacznie droższych.



CHARAKTERYSTYKA DRUKARKI

System pracy

Drukarka znakowo-mozaikowa (dot matrix) z ruchomą głowicą, drukującą również podczas powrotu.

Głowica drukarska

Zawiera 9 igieł ze stali wolframowej o średnicy 0,3 mm każda.

Rodzaj papieru

W drukarce można stosować papier z perforacją na lewej i prawej krawędzi o szerokości formatu A4 (format + perforacja) ułożony we wstędze typu ZIG-ZAG, przesuwany przez wałek wyposażony w tak zwany "traktor", ciągnący papier. Można również stosować papier w arkuszach formatu A4 (od maszyny do pisania), przesuwany przez normalny wałek gumowy będący jednocześnie zderzakiem dla igieł głowicy drukującej. Drukarka może być dodatkowo wyposażona w automat podający papier w arkuszach formatu A4.

Wysuw papieru

Tabulacja pionowa w dowolnym kierunku. Zależnie od ustawienia specjalnego wyłącznika – programowa lub automatyczna zmiana strony. Można ustawić wyłącznik w pozycji umożliwiającej drukowanie ciągłe (bez podziału na strony).

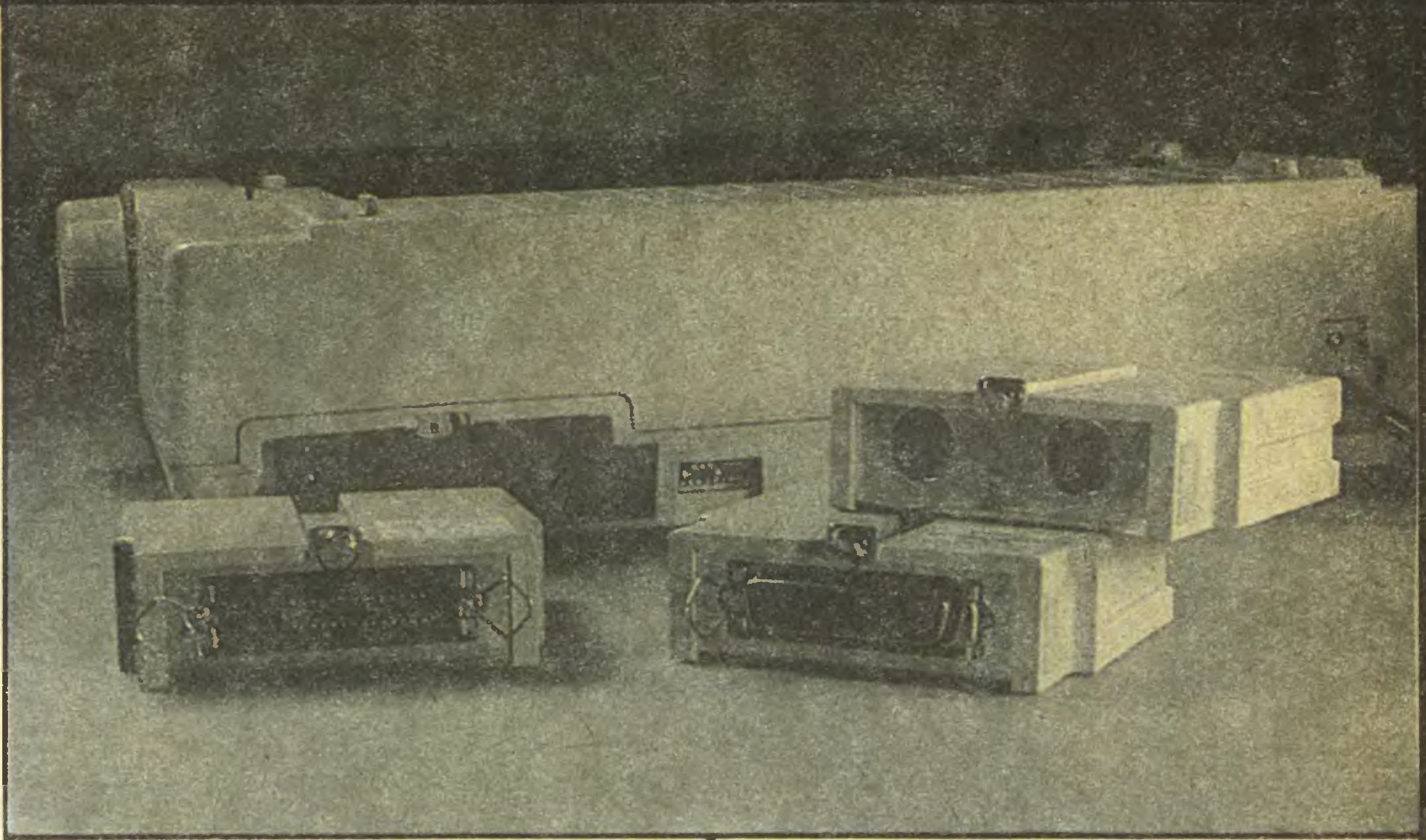
Szybkość druku

Tekst "normalny" drukowany jest z prędkością ok. 120 znaków na sekundę. Tekst w trybie NLQ (Near Letter Quality – tak zwany tryb korespondencyjny), gwarantującym wysoką jakość druku – ok. 30 znaków na sekundę.

Typ druku

Drukarka może drukować znaki normalne (matryca 9×11 punktów) lub znaki typu NLQ (matryca 18×23 punkty). Litery typu PICA i ELITE drukarka NL-10 może drukować jako normalne (80 znaków w wierszu na formacie A4), zwężone (96 znaków), zagęszczone (136 znaków), dwukrotnie poszerzone, dwu- lub czterokrotnie powiększone. Litery mogą być podkreślone, pisane kursywą, wytłuszczone. Możliwe jest również pisanie indeksów i wykładników potęg (sub- i superscript).

Drukarka umożliwia drukowanie liter w negatywie (litera pozostaje koloru papieru, a zadrukowywane jest tło), programowanie odstępu między wierszami od 1/6 do n/216 cala, drukowanie tekstu z odstępami proporcjonalnymi między literami, druk tekstu wyrównanego do prawej lub lewej strony arkusza papieru, a wreszcie "środkowanie" tekstu. Wszystkie rodzaje druku dostępne są programowo z komputera współpracującego z drukarką. W trybie normalnym możliwe są wszystkie kombinacje rodzajów druku.



Podłączenie do komputera

Drukarka STAR NL-10 wyposażona jest (w tylnej części) w gniazdo dla wymiennych modułów interfejsów, umożliwiających podłączenie drukarki do komputera. Drukarka dostarczona do naszej redakcji posiadała trzy moduły: interfejs do współpracy z komputerami IBM PC lub ich odpowiednikami, interfejs do współpracy z komputerami Commodore C64 i C128 oraz interfejs równoległy typu centronics do współpracy z większością komputerów domowych (np. Amstrad-Schneider, ZX Spectrum). Współpraca systemu mikrokomputerowego z drukarką nie nastręcza żadnych problemów. Z dołączonej do każdego interfejsu instrukcji można dość szybko poznać możliwości drukarki – zawiera ona podstawowe informacje poparte przykładowymi programami.

Test KOMPUTER'a

Redakcja Popularnego Miesięcznika
Informatycznego KOMPUTER

testuje drukarke mozaikowa STAR NL-10.
Drukarka NL-10 umożliwia drukowanie różnymi typami czcionek.
Możliwe sa podkreślenia a także drukowanie potęg i oczywiście indeksów. Każdy tekst może być drukowany czcionka ELITA lub normalnym drukiem PICA, można użyć też specjalnej czcionki Near Letter Quality w prawie idealny sposób naśladujacej czcionkę normalnej maszyny do pisania. Jeżeli jest to konieczne można także drukować drukiem proporcjonalnym. Oprócz NLQ wszystkie rodzaje pisma moga być miedzy soba dowolnie mieszane.

Drukarka: STAR NL-10 - Star Micronics Co.,Ltd. 1985
Komputer: ZX Spectrum - Sinclair Research Ltd. 1982
Program : Tekst/ed02 + NL10 - SP/M SPEKTRUM 1985

Sterowanie drukarką

Oprócz programowej współpracy z komputerem drukarka NL-10 posiada pulpit sterujący dla ręcznego ustawienia 8 funkcji. Sterowanie realizowane jest za pomocą sensorowych mikrowyłączników. Wszystkie realizowane funkcje sygnalizowane są diodami elektroluminescencyjnymi. Z pulpitu można ustawić lewy i prawy margines, wybrać rodzaj druku (normalny, zwężony, zagęszczony lub NLQ), spowodować wysuw papieru do nowej strony. Dodatkowymi przełącznikami w tylnej części drukarki można wyłączyć czujnik końca papieru, ustalić długość strony, wybrać zestaw znaków charakterystycznych dla alfabetu szwedzkiego, hiszpańskiego, duńskiego, niemieckiego, francuskiego, angielskiego, włoskiego i komputerów Commodore.

Inne możliwości

Dodatkowo drukarka NL-10 może przyjąć 96 znaków zaprogramowanych przez użytkownika (down load characters). Może również realizować znaki specjalne i znaki graficzne komputerów IBM PC lub ich odpowiedników. Drukarka pracuje także w trybie graficznym (kody wysyłane z komputera sterują działaniem poszczególnych igieł głowicy drukującej).

Zasilanie

Z sieci prądem zmiennym 220 V 50/60 Hz.

Wymiary

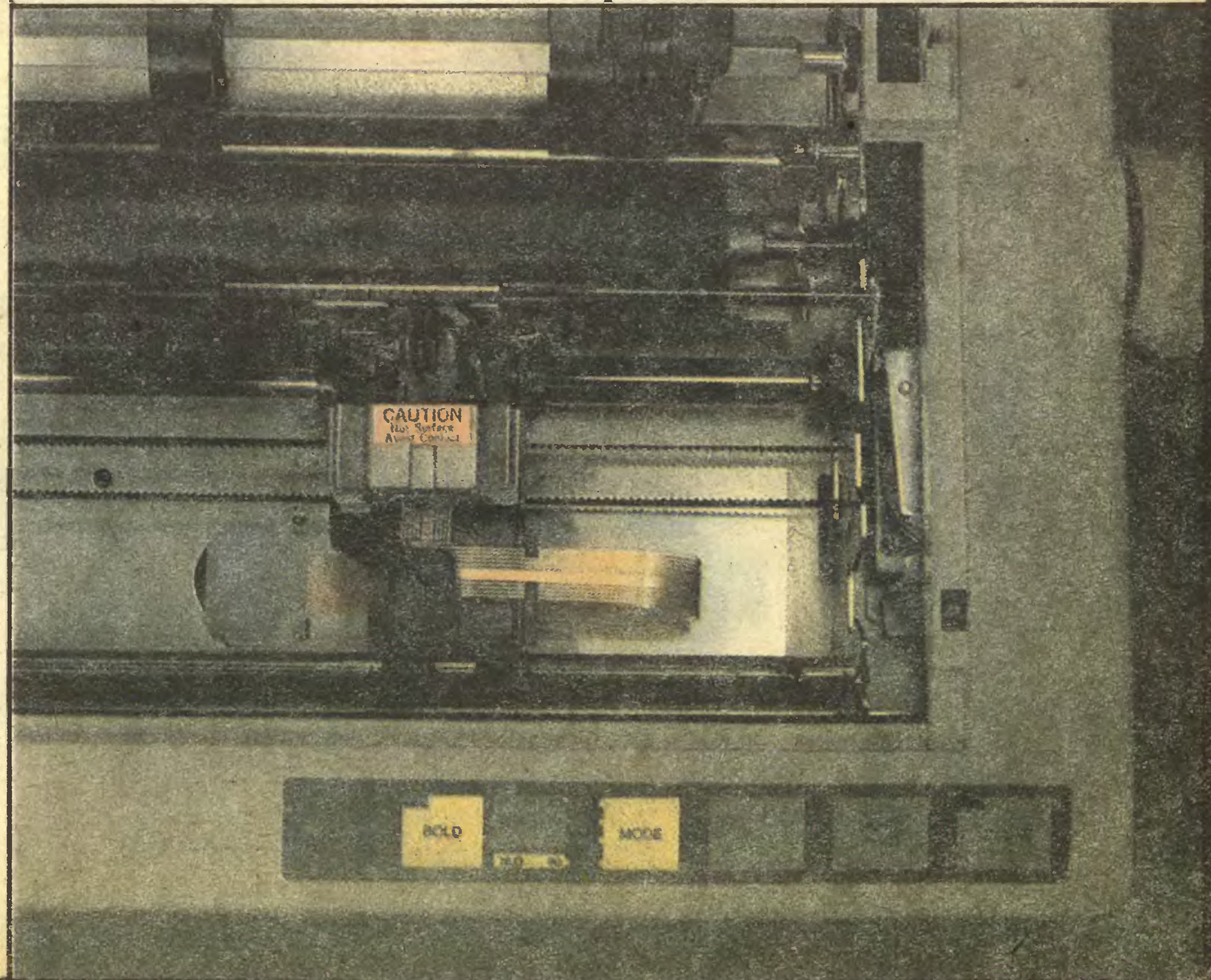
Długość 400 mm, szerokość 336 mm, wysokość 104 mm.

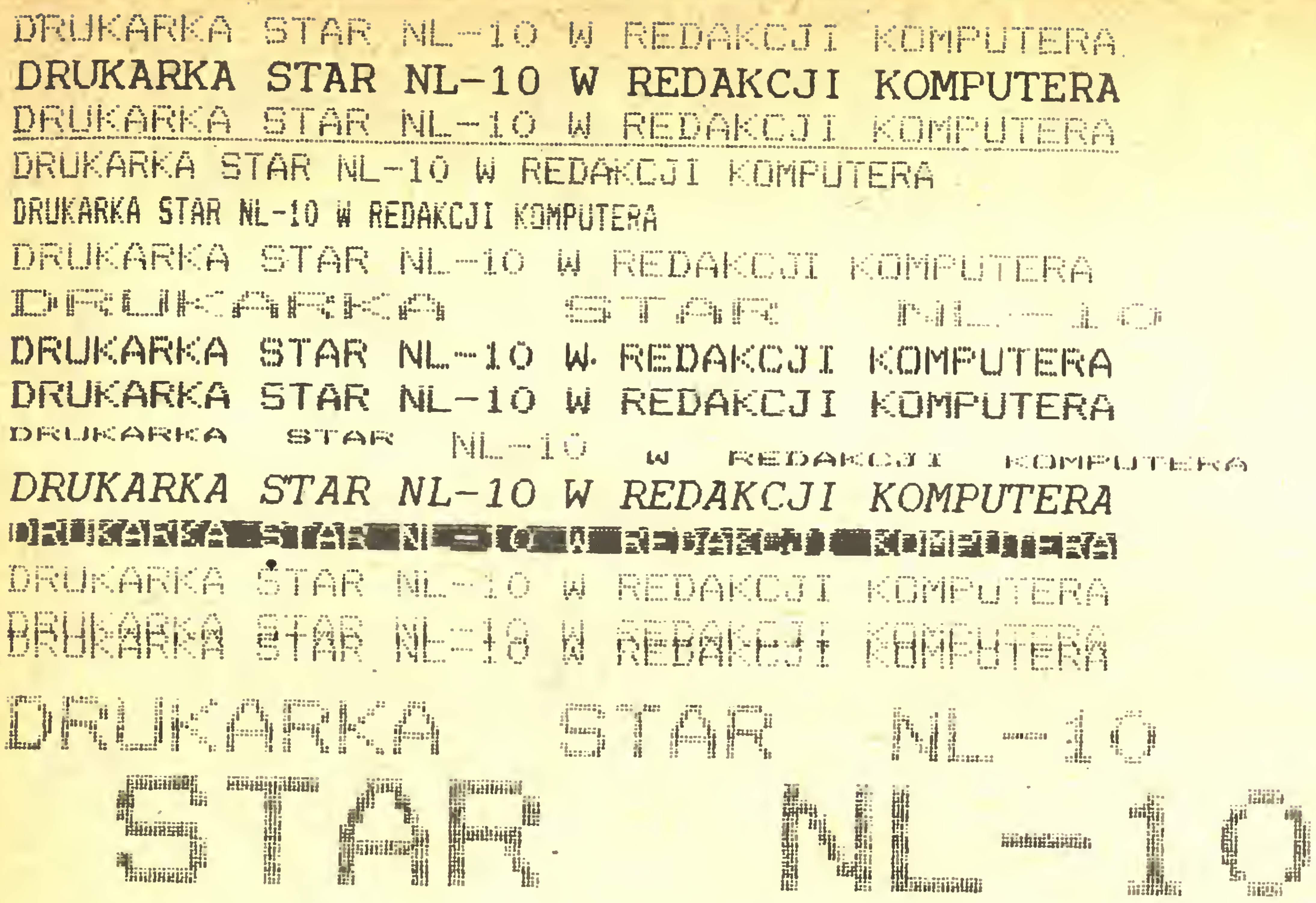
Waga

Masa kompletnej drukarki z zamontowanym interfejsem wynosi 6.6 kg.

TEST

Redakcja miała możliwość sprawdzić współpracę drukarki STAR NL-10 z komputerami Commodore C64. Olivetti M24 (dzięki uprzejmości Ryszarda Wojtkiewicza z Zakładu Wysokich Ciśnień PAN) i ZX Spectrum. Wszystkie komputery pracowały z drukarką bez zastrzeżeń. Możliwe było uzyskanie wydruku listingów, druk tekstu w dowolnym systemie lub mieszanie rodzajów druku. Jedynie w trybie NLQ (korespondencyjnym) przy współpracy z Commodore C64 niemożliwe jest mieszanie rodzajów druku. Drukarka bardzo dobrze odwzorowuje grafikę tworzoną przez komputer. Przenoszenie kolorowej grafiki z ekranu monitora na drukarkę pociąga za sobą gubienie niektórych kolorów. Gdy grafika tworzona jest w średniej lub małej rozdzielczości, to kolory repre-



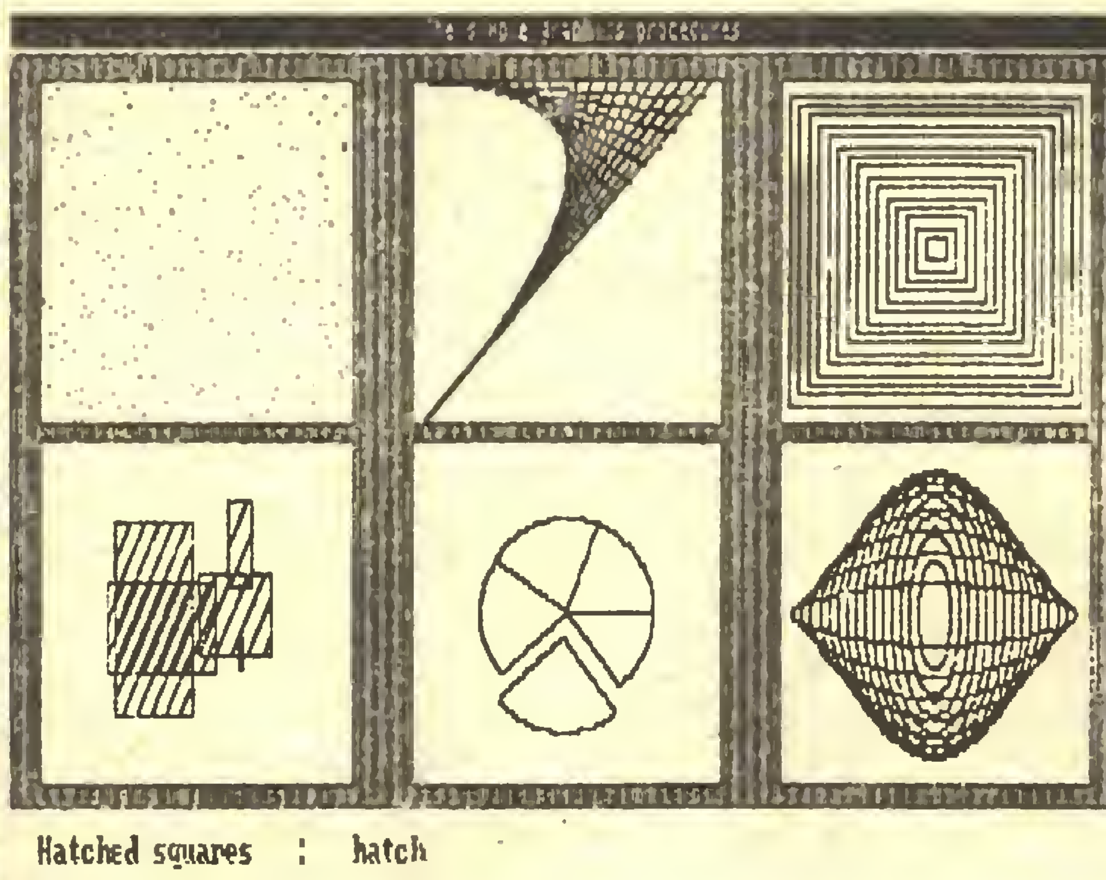


Współpraca z C-64

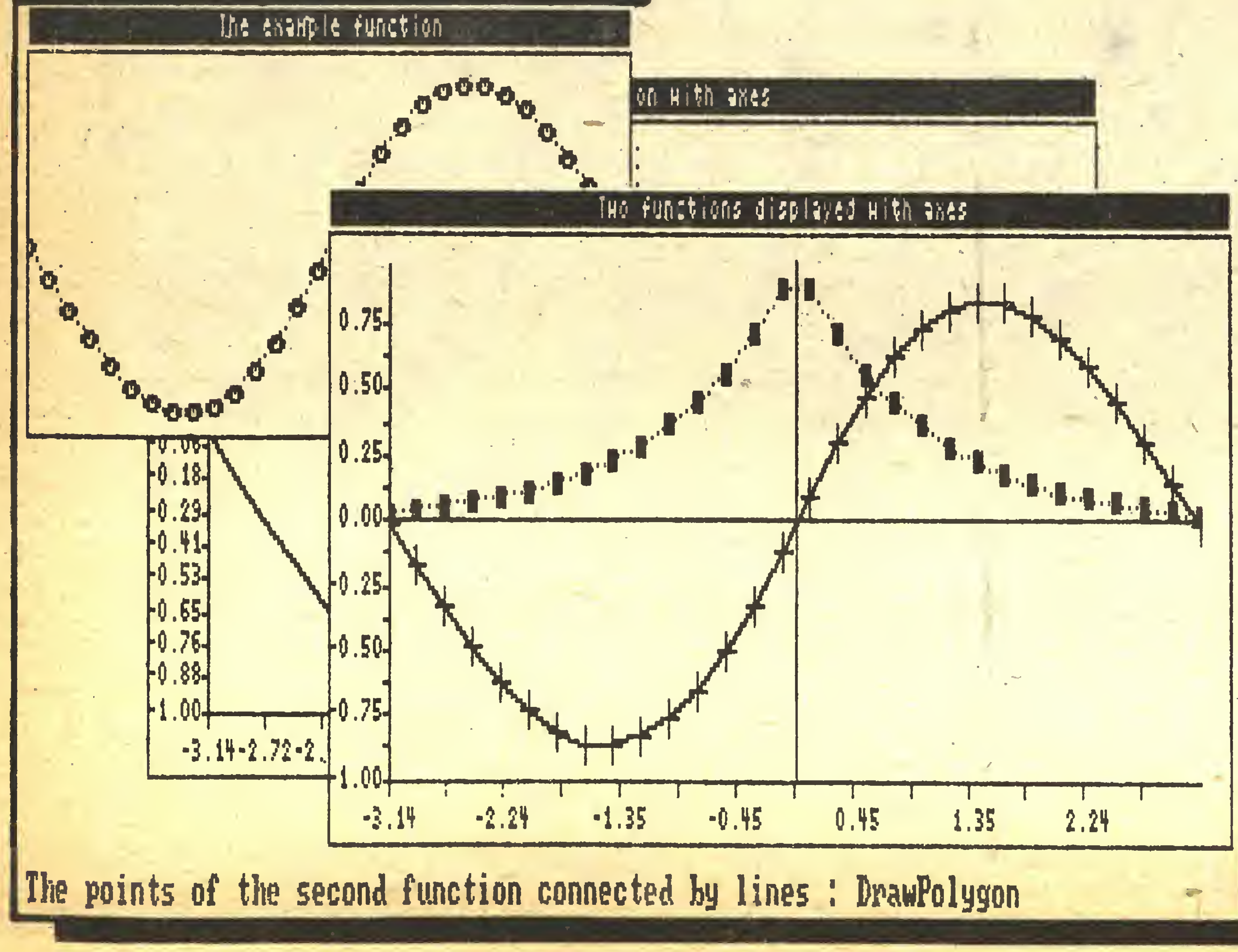
STAR NL -10 jest solidną drukarką mozaikową z dużym wyborem różnych możliwości, nie proponowanych dotychczas w tej klasie cenowej. Oferuje dobrą jakość druku, może być przyłączana – za pośrednictwem modułów interfejsu – do prawie wszystkich rodzajów komputerów. NL-10 jest wyposażona w półautomatyczne urządzenie do podawania pojedynczych arkuszy papieru, możliwa jest także prosta i szybka zmiana rodzaju druku.

Drukarka NL-10 jest dobrym pomocnikiem w codziennej korespondencji. Nawet wtedy, gdy trzeba trochę dłużej poczekać na lepszy wydruk (w trybie NLQ).

c't (Computer Technik) maj 1986



Współpraca z OLIVETTI M24



zentowane są na papierze jako różne odcienie szarości. Drukarka realizuje to przez silniejsze lub słabsze uderzenie igieł głowicy w taśmę barwiącą, co powoduje powstanie ciemniejszych lub jaśniejszych punktów na papierze. Gdy chcemy przenieść na papier grafikę o dużej rozdzielczości (640×400 punktów w 16 kolorach), to drukarka interpretuje tylko znaki kolorowe o kolorach nieparzystych (jako czarne punkty), inne kolory ignoruje. Sytuacja taka wystąpiła przy współpracy NL-10 z komputerem Olivetti M24. Grafika czarno-biała przenoszona była bez zarzutu.

W drukarce zastosowano kasetę z taśmą barwiącą. Taśma ma szerokość 13 mm i jest zwinięta w pętlę Möbiusa. Wymiana zużytej taśmy polega na wymianie całej kasety, co jest łatwe i szybkie, choć mało użyteczne na naszym rynku z braku takich kaset w sprzedaży. Majsterkowicze mogą wymienić taśmę z kasety samodzielnie. Operacja taka wymaga opracowania doświadczalnej metody łączenia taśmy od maszyny do pisania w pętlę bez końca, aby połączenie było trwałe i elastyczne. Próbę taką z pozytywnym wynikiem przeprowadziliśmy w redakcji (niezbędna lutownica, trochę kleju i cierpliwość).

Ciekawymi własnościami drukarki NL-10 jest funkcja "dump" i możliwość definiowania rozkazów "macro".

Funkcja "dump" umożliwia drukowanie tekstu wysyłanego przez komputer w formie listingu liczb heksadecymalnych i kodów ASCII. Daje to możliwość kontroli pracy drukarki oraz kontroli zaprogramowanych znaków użytkownika.

Definiowanie rozkazów "macro" polega na "schowaniu" pod jednym kodem takich operacji jak wybór rodzaju druku, ustawienie lewego i prawego marginesu, ustalenie wielkości odstępu między wierszami. Każdorazowe wywołanie rozkazu "macro" powoduje wykonanie wszystkich opisanych funkcji jednocześnie.

PODSUMOWANIE

STAR NL-10 to bardzo dobra, nowoczesna drukarka o szerokich możliwościach, współpracująca także z najpopularniejszymi i najtańszymi systemami mikrokomputerowymi. Zwiększa możliwości wykorzystania komputera przy wspomaganiu projektowania i pracach biurowych.

Podstawowe zalety:

- bogate możliwości drukarskie,
- praca w trybie graficznym,
- definiowany zestaw znaków,
- wymienne interfejsy,
- druk na papierze w arkuszach A4 bez perforacji.

Nieuniknione wady:

- taśma barwiąca w kasecie,
- częściowe ograniczenia możliwości przy współpracy z komputerami Commodore,
- mała prędkość druku w trybie NLQ,
- brak możliwości zwężania i poszerzania liter w trybie NLQ,
- na razie brak na naszym rynku.

ZENON RUDAK

PRZEDSIĘBIORSTWO ZAGRANICZNE

apina

ZAKŁAD ELEKTRONIKI

informuje PT Klientów, iż poczynając od dnia 2 maja 1986 roku serwis gwarancyjny i pogwarancyjny montowanych przez nas urządzeń mikrokomputerowych prowadzi w naszym imieniu autoryzowana firma:

infotech

al. Konstytucji 3 Maja 10, Zielona Góra

Pod tym też adresem należy przekazywać urządzenia do naprawy. Pozostałe warunki obsługi gwarancyjnej i pogwarancyjnej pozostają bez zmian. Firma ta ponadto świadczy w naszym imieniu usługi w zakresie wymiany klawiatury typu =standard= na twardą klawiaturę typu PLUS w mikrokomputerach ZX SPECTRUM.

Ponadto informujemy o naszych nowościach:

APC-16 to 16-bitowy komputer osobisty kompatybilny z **IBM PC**

Zapamiętaj ten symbol:

APC-16 to **APINA PERSONAL COMPUTER**

APC-16

to 16 bitów do Twojej dyspozycji
termin realizacji dostawy:
do 3 miesięcy od daty złożenia zamówienia

LIGHT PEN TURBO!!!

rewelacyjna przystawka do mikrokomputera ZX SPECTRUM. Bardzo bogate menu oraz ciekawe i absolutnie nowe rozwiązanie hardware'owe czynią to urządzenie konkurencyjnym do przystawki typu MOUSE.

Szczegółowe informacje dotyczące naszej oferty można uzyskać drogą telefoniczną lub teleksową.

Przedsiębiorstwo Zagraniczne
Zakład Elektroniki

apina

pl. Bohaterów Stalingradu 28
65-067 Zielona Góra
tel. 33-51 tlx: 0433266

BR-223

STOPER

Chyba każdy adept sztuki programowania po jednym czy dwóch miesiącach podejmuje trud wykonania programu "ZEGAREK". Program "STOPER" też można zaliczyć do tej klasy. Ma on jednak kilka cech podnoszących jego wartość. Zrealizowany został w kodzie maszynowym, co umożliwiło powiększenie cyfr ośmiokrotnie w stosunku do normalnej czcionki i uzyskanie dokładności kilku setnych części sekundy. Program jest wyposażony w funkcję zwykłego stopera oraz w dwie inne (międzyczasy i CONTINUE). Z tego powodu może być bardzo przydatny np. na lekcjach fizyki, na których przeprowadza się badania spadku ciała po równi pochyłej, rozładowania kondensatora, stygnięcia ciał w zależności od czasu, czy wykonuje inne doświadczenia.

Część w języku Basic można rozwinąć i poszerzyć o podprogramy rysujące wykresy lub tabele, obliczanie przyrostów czasu, wiążąc zapamiętane przez mikrokomputer międzyczasy z innymi wielkościami w zależności od potrzeb użytkownika.

Stoper uruchamia się klawiszem "S", a zatrzymuje przez naciśnięcie klawisza "K". Ponownie można go uruchomić naciskając "S" (zerowanie czasu) lub "C" (CONTINUE – kontynuacja odmierzania czasu). Międzyczasy można zapamiętać naciskając klawisz "N". Zostaną one automatycznie wypisane po powrocie do Basica, co następuje przez naciśnięcie klawisza "SPACE".

Aby przenieść program na taśmę magnetofonową, wprowadź do mikrokomputera ZX Spectrum (48) "PROGRAM 1" i nagraj go na taśmę komendą SAVE "STOPER" LINE 999Ø. Następnie wprowadź do komputera z klawiatury "PROGRAM 2", uruchom go komendą RUN 1ØØØ. Program ten wczyta dane zawarte w instrukcjach DATA (numery instrukcji od 1ØØ do 39Ø) i wprowadzi je do pamięci poczynając od adresu 56ØØØ jako kod maszynowy. (Program ten sprawdza też sumę każdych pierwszych dziesięciu danych w instrukcjach DATA).

Jeżeli nie popełniłeś błędu, na ekranie ukaże się informacja "START TAPE AND PRESS ANY KEY", do której należy się dostosować nagrywając na taśmę tuż za poprzednio tam umieszczonym programem "STOPER" część zasadniczą programu w kodzie maszynowym (bytes) pod nazwą "stopcode" (patrz instrukcja nr 1Ø8Ø). Gdy na skutek wykonania części instrukcji 1Ø9Ø pojawi się w najniższej linii ekranu napis "cofnij taśmę...", wystarczy tylko przewinąć taśmę do odpowiedniego miejsca i uruchomić magnetofon w celu weryfikacji. Instrukcja VERIFY ""CODE zawarta jest już w drugiej części linii 1Ø9Ø.

```
 1Ø RANDOMIZE USR 56ØØØ
 2Ø GO SUB 1ØØ: LET a$=":": LET
 a=563ØØ: LET l=1: PRINT TAB 1Ø;
"międzyczasy"''
 3Ø LET i=Ø: FOR n=a+7 TO a+14:
 LET i=i+PEEK n: NEXT n
 4Ø IF i=Ø THEN  GO SUB 1ØØ: GO
 TO 1Ø
 5Ø PRINT TAB 9-LEN STR$ l;"T";
l;PEEK a;a$;PEEK (a+1);PEEK (a+2
);a$;PEEK (a+3);PEEK (a+4);a$;PE
EK (a+5);PEEK (a+6)
 6Ø FOR n=a TO a+6: POKE n,Ø: N
EXT n
 7Ø LET a=a+7: LET l=l+1: GO TO
 3Ø
 1ØØ BEEP .2,3Ø: PRINT #1;AT 1,4
; FLASH 1;"naciśnij dowolny klaw
isz"
 11Ø IF INKEY$="" THEN  GO TO 11
Ø
 12Ø CLS : RETURN
999Ø LOAD ""CODE : POKE 23693,71
: BORDER Ø: CLS : GO TO 1Ø
```

Program 1

Jeżeli jesteś nieco rozczarowany wykorzystaniem w tym programie pisaniem kodów maszynowych za pomocą POKE–ów i ciekawi Cię, jak ten program powstawał w asemblerze, pozostaje Ci czekać na kolejny artykuł na ten temat lub... spróbować własnych sił!!!

Życzę powodzenia w uruchomieniu programu "STOPER", który może być świetnym prezentem dla Twojego nauczyciela fizyki, jeżeli szkoła posiada już upragnione Spectrum.

WOJTEK BIAŁEK
(uczeń klasy II B
LO Krzeszowice k/Krakowa)

```
 1ØØ DATA 33,236,219,34,2Ø7,219.
62,127,1,2,114Ø
 11Ø DATA 14,22,88,33,22Ø,219,94
,18,35,16,759
 12Ø DATA 251,2Ø,6,2,13,32,245,1
75,5Ø,218,1Ø12
 13Ø DATA 219,6,7,33,2Ø9,219,54,
Ø,35,16,798
 14Ø DATA 251,33,24,89,34,216,21
9,33,215,219,1333
 15Ø DATA 3Ø,7,213,229,17,121,61
,126,6Ø,2Ø3,1Ø67
 16Ø DATA 39,2Ø3,39,2Ø3,39.131.9
5,42,216,219,1226
 17Ø DATA 62,6,245,14,64,62,6,24
5,26,161,891
 18Ø DATA 32,4,54,64,24,2,54,127
,35,2Ø3,599
 19Ø DATA 57,241,61,32,238,14,26
,9,19,241,938
 2ØØ DATA 61,32,225,14,7,62,1Ø,4
2,216,219,888
 21Ø DATA 237,66,189,32,4,14,2,2
37,66,62,9Ø9
 22Ø DATA 25Ø,189,32,5,1,226,Ø,2
37,66,34,1Ø4Ø
 23Ø DATA 216,219,225,43,2Ø9,29,
32,17Ø,33,215,1391
 24Ø DATA 219,1,2ØØ,219,1Ø,52,19
Ø,32,5,3,931
 25Ø DATA 115,43,24,246,1,Ø,Ø,25
,13,32,499
 26Ø DATA 252,237,91,2Ø7,219,33,
2Ø9,219,6,7,148Ø
 27Ø DATA 126,35,18,19,16,25Ø,62
,127,219,254,1126
 28Ø DATA 23Ø,1,2ØØ,58,218,219,2
54,255,4Ø,24,1499
 29Ø DATA 62,253,219,254,23Ø,2,3
2,4,47,195,1298
 3ØØ DATA 22Ø,218,62,254,219,254
,23Ø,8,32,222,1719
 31Ø DATA 47,5Ø,218,219,62,127,2
19,254,23Ø,8,1434
 32Ø DATA 32,28,58,219,219,184,3
2,18,1,7,798
 33Ø DATA Ø,42,2Ø7,219,9,34,2Ø7,
219,62,16,1Ø15
 34Ø DATA 5Ø,219,219,195,233,218
,33,219,219,53,1658
 35Ø DATA 62,191,219,254,23Ø,4,3
2,241,5Ø,218,15Ø1
 36Ø DATA 219,24,221,221,1Ø,1Ø,1
Ø,6,1Ø,6,737
 37Ø DATA 1Ø,236,219,Ø,Ø,Ø,Ø,Ø,Ø
,1,466
 38Ø DATA 1,88,Ø,11,2,33,34,35,6
6,69,339
 39Ø DATA 7Ø,98,13Ø,133,134,163,
79,175,79,175,1236
1ØØØ LET a=56ØØØ
1Ø1Ø FOR n=Ø TO 29
1Ø2Ø LET s=Ø
1Ø3Ø FOR m=Ø TO 9
1Ø4Ø READ b: LET s=s+b: POKE a+1
Ø*n+m,b
1Ø5Ø NEXT m: READ i: IF s<>i THE
N  PRINT "błąd w linii ";1ØØ+n*1
Ø: STOP
1Ø7Ø NEXT n
1Ø8Ø SAVE "stopcod"CODE 56ØØØ,3Ø
Ø
1Ø9Ø PRINT #1;"cofnij taśmę w ce
lu weryfikacji": VERIFY ""CODE
```

6502 – przykłady procedur

W ostatniej części cyklu na temat 6502 zostaną przedstawione przykładowe programy dla tego mikroprocesora. Niektóre z nich zawierają typowe chwyty stosowane przez producentów oprogramowania dla Apple, Commodore lub Atari. Nie należy zatem traktować ich jako ćwiczeń wprowadzających do ogólnych zagadnień programowania w języku asemblera. Przykłady zostały dobrane z myślą o czytelniku mającym pewne doświadczenie w programowaniu mikroprocesorów. Stąd brak wiadomości na temat podstawowych operacji wielokrotnej precyzji, obsługi urządzeń zewnętrznych i innych procedur, przy których obowiązują zasady jednakowe dla wszystkich mikroprocesorów uniwersalnych.

● Na początek – typowa procedura przesłania niewielkiego bloku z jednego miejsca pamięci do drugiego (PROCEDURA 1). Występuje z niej charakterystyczne dla 6502 wykorzystanie 8-bitowego rejestru indeksowego, którego zmienna zawartość jest dodawana do 16-bitowego stałego adresu. Rejestr indeksowy jest równocześnie wykorzystywany jako licznik przesyłanych bajtów. Dla porównania warto przytoczyć indeksowy tryb adresowania dla Z 80, którego rejestry indeksowe są 16-bitowe. Adres efektywny jest w nim obliczany przez sumowanie zmiennej, 16-bitowej zawartości rejestru indeksowego i stałego, 8-bitowego przesunięcia, co wymaga zupełnie innej techniki korzystania z rejestrów indeksowych.

```
* Procedura przeslania bloku (do 128 bajtow)
*  w kierunku wiekszych\ adresow w pamieci
*
Transfer   LDA $ile-1
Powtarzaj  LDA skad,X
           STA dokad,x
           DEX
           BNE Powtarzaj
           RTS
```

● PROCEDURA 2 przedstawia mnożenie dwóch liczb 8-bitowych. Należy zwrócić uwagę na niezbędne zerowanie wskaźnika przeniesienia CARRY (instrukcją CLC) każdorazowo przed dodawaniem. Wynika to z braku instrukcji dodawania bez CARRY. Podobnie jest w przypadku odejmowania, przed którym CARRY należy zawsze ustawiać.

```
* Mnozenie 8 * 8 bitow
* Dane w naturalnym kodzie binarnym
*
Mnozenie   LDA $0
           LDX $8         Licznik bitow
Powtarzaj  LSR Czynnik1
           BCC Przeskocz gdy bit=0
           CLC
           ADC Czynnik2
Przeskocz  ROR
           ROR Iloczyn
           DEX
           BNE Powtarzaj
           STA Iloczyn+1
           RTS
```

● Algorytmy konwersji kodów reprezentacji liczb – z binarnych na dziesiętne i odwrotnie – są przedstawione w postaci PROCEDURY 3 i PROCEDURY 4. Pierwsza z nich wykorzystuje bardzo wygodny w kodowaniu algorytmów, dziesiętny tryb 6502. Obie procedury zostały zaprojektowane dla danych 8-bitowych (w reprezentacji dwójkowej), ale jest możliwe ich łatwe rozszerzenie przez zwielokrotnienie odpowiednich operacji.

```
* Konwersja 1-bajtowej liczby binarnej
*  na 2-bajtowa liczbe w kodzie BCD
*
BCD.Konw   LDX $8            Licznik bitow
           LDA $0
           STA Wynik
           STA Wynik+1
           SED               Tryb dziesietny
*
Powtarzaj  ROL Liczba.binarna
           BCC Przeskocz gdy bit=0
           INC Wynik
           BNE Przeskocz
           INC Wynik+1
           DEX
           BEQ Koniec
*
Przeskocz  CLC               --
           LDA Wynik           !
           ADC Wynik           ! Dziesietne
           STA Wynik           ! mnozenie
           LDA Wynik+1         !  przez 2
           ADC Wynik+1         !
           STA Wynik+1       --
           JMP Powtarzaj
*
Koniec     CLD
           RTS
* Konwersja 3-cyfrowej (2-bajtowej) liczby
*  w kodzie BCD, mniejszej od 256, na
*  1-bajtowa liczbe binarna
*
Bin.konw   LDA Liczba.BCD+1  Pierwsza cyfra
           JSR Razy.10
           LDA Liczba.BCD
           LSR
           LSR
           LSR
           LSR               Druga cyfra
           CLC
           ADC Wynik
           JSR Razy.10
           LDA Liczba.BCD
           AND $$0F          Trzecia cyfra
           CLC
           ADC Wynik
           STA Wynik
           RTS
*
Razy.10    STA Wynik
           ASL               x 2
           ASL               x 4
           ADC Wynik         x 5
           ASL               x 10
           STA Wynik
           RTS
```

● Przy projektowaniu oprogramowania często napotyka się na zagadnienie rozpoznawania znaków lub ogólnie – elementów kodu i przekazywania sterowania do odpowiadających im procedur roboczych. Przykładem może być choćby interpretacja programu w Basicu. Typowe rozwiązanie problemu tego rodzaju zawiera PROCEDURA 5. W pętli SZUKAJ znak w akumulatorze jest kolejno porównywany z elementami w tablicy wzorców. Po odnalezieniu wzorca jego indeks jest dwukrotnie zwiększany, co daje w wyniku indeks adresu odpowiadającej mu procedury w tablicy skoków. Adres jest umieszczany na stosie w ten sposób, by wykonanie instrukcji RTS spowodowało właściwy skok.

Jest to trik mało czytelny i niewiele efektywniejszy od użycia skoku pośredniego, lecz wyjątkowo często stosowany przez producentów oprogramowania systemowego dla 6502. Aby określić, gdzie w pamięci przebywa kod rozpoznawania np. elementów języka i odpowiednie tablice, najczęściej wystarczy przeszukać (oczywiście maszynowo!) obszar, w którym rezyduje oprogramowanie i odnaleźć sekwencję PHA, RTS – $48, $60.

```
* Procedura rozpoznajaca znak w akumulatorze
*  i przekazujaca sterowanie do odpowia-
*  dajacego mu podprogramu obslugi
*
Rozpoznaj  LDY $Ilosc.wzorcow-1
Szukaj     CMP Tablica.wzorcow,Y
           BEQ Skocz
           DEY
           BPL Szukaj   gdy licznik >= 0
           JMP Brak.wzorca  w tablicy
*
Skocz      TYA              Pomnoz
           ASL               numer wzorca w Y
           TAY               przez 2
           LDA Tablica.skokow+1,Y
           PHA
           LDA Tablica.skokow,Y
           PHA
           RTS              Skocz pod adres
*                            pobrany ze stosu
Tablica.wzorcow
           .DA $Wzorzec1,$Wzorzec2, ... ,$WzorzecN
Ilosc.wzorcow .EQ *-Tablica.wzorcow
Tablica.skokow
           .DA Skok1-1,Skok2-1, ... SkokN-1
```

Należy zwrócić uwagę na to, że adresy w tablicy skoków są zmniejszone o 1. Wynika to stąd, że podczas wykonywania instrukcji RTS pobierany ze stosu adres jest zwiększany o 1.

● PROCEDURY 6 i 7 są przeznaczone dla konstruktorów oprogramowania, zainteresowanych zakładaniem pułapek w uruchamianym programie. Pierwsza z nich – ZAPAMIĘTAJ zapamiętuje zawartości rejestrów procesora i przekazuje sterowanie do PROGRAMU-MONITORA. Przy jego użyciu można przejrzeć i zmodyfikować zapamiętane zawartości rejestrów. Można też skorzystać z innych funkcji oferowanych przez firmowy monitor, który najczęściej właśnie jest pozbawiony możliwości zakładania pułapek. Adres procedury ZAPAMIĘTAJ należy podstawić do wektora obsługi przerwań, który zazwyczaj jest udostępniony użytkownikowi komputera.

W celu przerwania programu w żądanym miejscu należy przed jego uruchomieniem podstawić tam kod instrukcji BRK, czyli po prostu zero.

```
* Procedura zapamietujaca zawartosci wszystkich
*  rejestrow dostepnych dla programisty
* Wejscie - po przerwaniu, lub BRK
*
Zapamietaj STA Akumulator
           STX X.rejestr
           STY y.rejestr
           PLA              Wskazniki ze stosu
           STA Wskazniki
           PLA              Mniej znaczacy bajt PC+2
           SEC
           SBC #2
           STA Licznik.rozkazow
           PLA              Bardziej znaczacy bajt PC+2
           SBC #0
           STA Licznik.rozkazow+1
           TSX
           STX Wskaznik.stosu
           JMP Porgram.monitor
```

Procedura ODTWÓRZ ładuje do rejestrów zapamiętane wcześniej lub zmodyfikowane wartości i przekazuje sterowanie do uruchamianego programu w miejsce wskazywane przez licznik rozkazów PC. Jeżeli była tam ustawiona pułapka – BRK, to należy ją wcześniej skasować – przez podstawienie kodu pierwotnej instrukcji. Oczywiście można to zautomatyzować przez rozbudowanie przedstawionych procedur, a nawet użyć je w poważnym programie uruchomieniowym.

```
* Procedura odtwarzajaca zawartosci wszystkich
*  rejestrow dostepnych dla programisty
*  i przekzujaca sterowanie do przerwa-
*  nego programu
*
Odtworz    LDX Wskaznik.stosu
           TXS
           LDY Licznik.rozkazow+1
           LDX Licznik.rozkazow
           BNE Dalej
           DEY
Dalej      DEX
           TYA
           PHA              Bardziej znaczacy bajt PC-1
           TXA
           PHA              Mniej znaczacy bajt PC-1
           LDA Wskazniki
           PHA              Wskazniki na stos
           LDY Y.rejestr
           LDX X.rejestr
           LDA Akumulator
           RTI
```

W procedurze ZAPAMIĘTAJ zawartość licznika rozkazów jest zwiększana o 2. Wynika to z równoczesnego w 6502 wykonywania rozkazu i pobierania następnego kodu. Po wykonaniu instrukcji BRK licznik rozkazów pozostaje "na wyrost" zwiększony o 2.

* * *

Przedstawione procedury stanowią jedynie wstęp do zbioru ćwiczeń dla programisty chcącego nauczyć się programowania 6502. Więcej informacji i przykładów można znaleźć w wielu podręcznikach programowania tego procesora. Szczególnie godna uwagi jest książka Leo J. Scanlona "6502 Software Design", Howard W. Sams and Co., Inc., USA 1983.

Na zakończenie cyklu o 6502 warto przypomnieć jego podstawową zaletę – jest bardzo łatwy i efektywny w programowaniu dzięki znakomicie zaprojektowanej organizacji wewnętrznej i równie dobrej liście rozkazów. Najlepiej świadczy o tym fakt, że mimo zwiększającej się sprzedaży mikroprocesorów 16- i 32-bitowych, 6502 bez kłopotów utrzymuje się na rynku. Są również konstruowane nowe komputery zbudowane w oparciu o 6502, także klasy profesjonalnej. Przykładem tych ostatnich może być LASER 128, w pełni zgodny z APPLE IIe, mało jeszcze u nas znany, a szkoda, gdyż – choć jest on 8-bitowy – ilość i jakość oferowanego dla niego oprogramowania nie ustępuje programom dla IBM PC, a LASER 128 jest znacznie od IBM PC tańszy.

Szczegóły o LASERZE – w jednym z najbliższych numerów "Komputera".

MACIEJ KASPERSKI

Dwa światy pod wspólnym dachem

Mikroprocesory firmy INTEL od ponad dziesięciu lat nadają ton rynkowi komputerów osobistych. Dominujący wśród ośmiobitowych – system operacyjny CP/M80 – opiera się o INTEL 8080. Zastosowanie przez firmę IBM w jej pierwszych mikrokomputerach PC i XT procesora INTEL 8080 zapewniło mu – wraz z bliźniaczym 8086 – dominującą pozycję w świecie szesnastobitowych.

INTEL 8088 jest zgodny programowo z 8086, różnice mają charakter wyłącznie sprzętowy (8088 ma ośmiobitową magistralę danych, zaś 8086 – szesnastobitową, dlatego działa szybciej). Sukces rodziny 8088/6 wynikał w dużym stopniu z łatwości adaptacji programów napisanych dla 8080. Firma INTEL dostarczyła nawet pakiet programów, automatycznie tłumaczących kod 8080 na 8088/6. Aby z nich skorzystać, należało wszakże dysponować źródłowym tekstem programu w języku asemblera. Przeciętny użytkownik kupuje jednak programy gotowe w postaci kodu maszynowego, a więc nie może samodzielnie przenieść swego starego oprogramowania z wysłużonego komputera klasy CP/M 80 na nowo nabyty IBM PC lub jego tajwańską kopię.

Wiele popularnych programów, stworzonych dla CP/M 80, zostało przez ich producentów adaptowanych dla IBM PC. Adaptacje te na ogół nie grzeszą jednak efektywnością. Oprogramowanie przeniesione „żywcem" z 8080 na 8088 nie wykorzystuje najsilniejszych stron nowego procesora. Przykładem niech będzie MICROSOFT-BASIC (w CP/M 80: MBASIC, w MS-DOS: GW-BASIC), który na niektórych komputerach 8-bitowych działa szybciej niż na IBM PC/XT. Adaptacja nie prowadzi więc do specjalnego podniesienia walorów programu i jest zabiegiem kosztownym.

W systemie CP/M 80 dostępne są tysiące wartościowych programów, których nie opłaca się przerabiać dla procesora 8088. Dotyczy to zwłaszcza oprogramowania oryginalnego, opracowanego dla pojedynczych użytkowników do konkretnych zastosowań.

Bardzo interesujące i skuteczne rozwiązanie przedstawionych bolączek zaproponowała japońska renomowana firma mikroelektroniczna NEC. Opracowała ona rodzinę nowych mikroprocesorów, spokrewnionych zarówno z 8080, jak i z szesnastobitowymi procesorami marki INTEL. Noszą one umowne nazwy V20-V50. Z punktu widzenia dzisiejszego użytkownika szczególnie atrakcyjne są dwa pierwsze: V20 (μPD70108) i V30 (μPD70116), pomyślane jako dość ścisłe odpowiedniki 8088 i 8086.

Zgodność (kompatybilność) z układami 8088/6 zachowano zarówno pod względem programowym, jak i elektrycznym. Wystarczy wyjąć z podstawki 8088 lub 8086, wstawić V20 lub V30 i komputer pracuje dalej, jak gdyby nigdy nic. Współpraca z koprocesorem arytmetycznym INTEL 8087 także przebiega bez zarzutu, jak w przypadku mikroprocesorów oryginalnych.

Mimo zgodności, V20 i V30 nie są jednak kopiami 8088/6. Wytwarzane są technologią CMOS, co znacznie zmniejsza pobór mocy (8086 pobiera ok. 1.7 W, V30 – tylko 0.5 W). Ich architektura wewnętrzna jest bardziej rozbudowana, co zapewnia wydajniejszą pracę przy tej samej co w 8088/6 częstotliwości zegara (taktowania). Wiele operacji, które 8088/6 wykonują przy pomocy mikroprogramu, V20 realizuje na drodze sprzętowej. Za przykład niech posłużą rozkazy przesunięć zawartości rejestrów o podaną liczbę bitów. 8086 potrzebuje na przesunięcie pierwszego bitu 9 cykli zegara i po 4 cykle na każdy następny bit. V30 potrzebuje na pierwszy bit 8 cykli plus 1 cykl na każdy kolejny. W rozkazach mnożenia i dzielenia zysk czasu jest kilkakrotny.

Lista rozkazów V20/30 została w stosunku do 8088/6 rozszerzona zarówno o rozkazy wprowadzone w procesorach INTEL 80188/6, jak i o szereg nowych rozkazów, usprawniających m.in. operacje na pojedynczych bitach i na liczbach w formacie BCD (dziesiętnym, kodowanym dwójkowo).

Najdonioślejszą własnością nowych procesorów jest jednak możliwość ich programowego "przełączenia" w tryb emulacji ośmiobitowego 8080. Układy V20/30 zachowują się tak, jak gdyby zawierały dwa niezależne procesory: 8088/6 oraz 8080. W danej chwili może oczywiście pracować tylko jeden z nich. Koncepcja komputera z dwoma procesorami nie jest nowa (np. Apple II, C 128), jednak zawsze opierała się o fizyczną obecność dwóch różnych procesorów, np. 6502 i Z 80.

W odróżnieniu od programowej emulacji 8080, zwalniającej tempo pracy prawie o rząd wielkości, rozwiązanie zastosowane w V20/30 jest czysto sprzętowe. „Wbudowany 8080" pracuje z pełną prędkością oryginalnego mikroprocesora, a nawet szybciej – zależnie od częstotliwości taktowania. Wielu niezależnych producentów wytwarza dziś odpowiedniki układów 8080 i

Procesory 8088/6 mogą adresować pamięć do 1 megabajta (2120 bajtów), mimo że ich szesnastobitowy licznik rozkazów PC może przyjąć największą wartość 2116-1=65535. Efektywny adres rozkazu jest bowiem sumą PC oraz pomnożonej przez 16 zawartości szesnastobitowego rejestru segmentu CS.

Niech rejestr CS zawiera 256, zaś OC – 9. Ostateczny adres, wyznaczony ze wzoru: ADR = 16 CS+PC wyniesie: 15 256+9=4105

Wynika stąd, że ten sam adres można przedstawić różnymi kombinacjami wartości CS i PC. Dla CS=0 i PC=4105 adresem efektywnym jest także 4105.

8088/6. Mówi się o nich, że są zgodne (kompatybilne) z wyrobami marki INTEL. Nowe procesory firmy NEC zyskały sobie żartobliwy, lecz uzasadniony przydomek "nadkompatybilnych".

Po wymianie 8088/6 na V20/30 nic nie stoi na przeszkodzie, aby do pamięci operacyjnej IBM PC lub podobnego komputera załadować program maszynowy z systemu CP/M, po czym "przełączyć" procesor w tryb 8080 i wykonać ten program jak w oryginalnym komputerze ośmiobitowym. Można sobie wyobrazić i takie rozwiązanie, w którym program napisany dla 8088/6 korzysta z posiadanych już i sprawdzonych procedur w języku wewnętrznym 8080.

V20 i V30 emulują tylko INTEL 8080, lecz nie Z 80, stosowany w nowszym sprzęcie ośmiobitowym. Nie stanowi to jednak istotnego ograniczenia. Ogromną większość oprogramowania w systemie CP/M napisano – w celu poszerzenia kręgu użytkowników – pod kątem 8080, nie stosując rozkazów Z 80. Jednym z nielicznych (na szczęście) wyjątków jest świetny PASCAL TURBO, dostępny jednak także dla IBM PC.

Jak odbywa się „przełączanie" procesorów? Wykorzystywany jest tu typowy dla 8088/6 mechanizm przerwań. Przypomnijmy, że po zaakceptowaniu przerwania procesory 8088/6 przechowują na stosie rejestr stanu F, rejestr segmentu CS i licznik programu PC, a następnie skaczą do procedury obsługi przerwania. Możliwych jest 256 różnych przyczyn przerwania i tyleż programów obsługi. Adresy procedur obsługi przerwań o numerach 0-255 (tzw. wektory przerwania) zapisane są w pierwszym kilobajcie pamięci operacyjnej: nr 0 w bajtach 0-3, nr 1 w bajtach 4-7 itd. Każdy adres obejmuje dwie liczby szesnastobitowe, przedstawiające nową zawartość rejestru segmentu CS i licznika programu PC.

V20/30 mają specjalny, niespotykany w 8088/6, rozkaz BRKEM (Break for emulation), wywołujący przerwanie programowe, rozpoczynające emulację 8080. Rozkaz BRKEM ma jednobajtowy argument, wskazujący na jeden spośród 256 wektorów przerwań. Po skoku do podanego adresu procesor emuluje już 8080.

Oryginalny INTER 8080 „widzi" tylko 64 KB, podczas gdy pamięć operacyjna IBM PC może liczyć 640 KB, (w pewnych warunkach – więcej). Kod maszynowy 8080 można umieścić w dowolnym obszarze pamięci. Programy w systemie CP/M 80 zaczynają się zwykle od adresu 256. Rozpoczynając emulację, wystarczy wpisać do rejestru segmentu taką wartość, aby przy liczniku programu PC zawierającym 256 adres efektywny był adresem początku programu w PAO. Przez to, i dzięki możliwości rozpoczynania emulacji od różnych adresów (różne argumenty BRKEM), w pamięci mogą równocześnie znajdować się i być na przemian wykonywane programy, które w oryginalnym komputerze 8-bitowym okupowały te same adresy i mogły być ładowane tylko na przemian.

Z trybu emulacji można w każdej chwili powrócić do naturalnego trybu szesnastobitowego 8088/6. Emulowany 8080 ma nieznany w oryginalnym procesorze rozkaz RETEM (Return from emulation), kończący emulację i powodujący powrót do programu maszynowego 8088/6, który emulację wywołał. Odbywa się to podobnie jak po zakończeniu obsługi przerwania.

Godny uwagi jest fakt, że wykonywany w trybie emulacji program dla 8080 może w każdej chwili wywołać podprogram w kodzie 8088/6! Rozkaz CALLN, nie występujący w INTEL, wywołuje tryb 8088/6 na zasadzie podobnej do przerwania programowego. Najbliższy rozkaz powrotu z przerwania przywróci ponownie tryb emulacji. Pozwala to programowi języku 8080 posiłkować się np. wywołaniami systemu operacyjnego dla procesora szesnastobitowego.

Przedstawiony mechanizm bardzo upraszcza wykonywanie na IBM PC programów stworzonych w systemie CP/M 80. Wystarczy po prostu użyć systemu operacyjnego CP/M 86, którego formaty zbiorów są zgodne z CP/M 86, a wywołania systemu operacyjnego bardzo przypominają CP/M 80.

Aby wykonać program stworzony dla systemu CP/M 80, trzeba stworzyć mu analogiczne jak w CP/M 80 otoczenie programowe. Dotyczy to zwłaszcza możliwości korzystania z usług modułów BDOS i BIOS. Nie trzeba jednak tworzyć tych modułów od podstaw. Wystarczy skonstruować ogólny szkielet, rozpoznający poszczególne wywołania, zaś "czarną robotę" pozostawić systemowi CP/M 86, wywołując jego poszczególne procedury. Kompletny program umożliwiający emulację systemu CP/M może liczyć zaledwie kilkaset bajtów.

Procesory V40 i V50 są w założeniu odpowiednikami rodziny INTEL 80188/6. Nie ma już jednak odpowiedniości elektrycznej, pozwalającej na prostą wymianę układów scalonych. Także i V40/50 mogą bez kłopotów emulować 8080. Tej ostatniej możliwości nie mają natomiast dwaj ostatni członkowie rodziny nowych procesorów NEC, noszący oznaczenia V60 i V70. Nie byłoby to zresztą sensowne: obydwa procesory należą już do generacji 32-bitowej.

Procesory V20 i V30 są pod względem cenowym porównywalne z oryginalnymi 8088/6. Procesory te mogą stanowić nadzieję na rozwiązanie problemu przenośności oprogramowania dla wielu użytkowników mikrokomputerów także i w Polsce. Z jednej strony jeszcze przez wiele lat będziemy mieli w kraju do czynienia z systemami ośmiobitowymi, z drugiej zaś standardem, powoli lecz wyraźnie, staje się IBM PC. Być może łatwiej będzie podjąć dziś decyzję o zakupie tańszego komputera ośmiobitowego, mając widoki na dalszą eksploatację opracowanego dla niego oprogramowania w przyszłości, po "przesiadce" na komputer szesnastobitowy.

ROLAND WACŁAWEK

\# \# \#

NOWA TECHNOLOGIA PAMIĘCI RAM

Bardzo silna handlowa presja firm japońskich, zalewających rynki światowe różnymi "kośćiami" spowodowała, że wiele elektronicznych firm amerykańskich wycofuje się z rynku układów ogólnego zastosowania, przechodząc na wytwarzanie "kości" na zamówienie, zaspokajających specjalne potrzeby. Nie wszystkie jednak – Texas Instruments postanowiła wytrzymać japońskie natarcie. Ufa w swoją superbroń – nową technologię wytwarzania układów. Polega ona na wytrawianiu mikroskopijnych rowków na poszczególnych płatkach krzemu. Zwiększa to ich powierzchnię, a umieszczanie tranzystorów i innych elementów układu w owych rowkach umożliwia uzyskanie większej gęstości upakowania.

Jest to o tyle ważne, że im mniejsza "kość", tym mniej kosztuje produkcja. Teksańczycy sporządzili już prototypową pamięć (o dostępie swobodnym) o pojemności 4 mln bitów – czyli cztery razy większą niż największe pamięci wprowadzane dopiero na rynek, przy czym ma ona znacznie mniejsze wymiary niż konkurencyjna pamięć 4-megabitowa firmy Toshiba Corp.

Texas Instruments upiera się przy produkowaniu pamięci o dostępie swobodnym, bowiem twierdzi, że na tych układach zawsze dotąd testowano nowe techniki i zarzucenie prac w tej dziedzinie uważa za poddanie się w całej sferze wyrobów o najwyższej jakości. (JAL)

NOWY C64

Commodore C65 już od kilku lat znajduje się na czele listy najpopularniejszych komputerów domowych. Zdobywał także tytuł komputera roku nadawany przez kilka czasopism mikrokomputerowych. W ubiegłym roku z każdych 3 sprzedanych w RFN komputerów domowych, dwa to C64. Nic więc dziwnego, że Commodore wcale nie zamierza zaprzestać produkcji C64, mimo że jest to już konstrukcja nieco przestarzała. Bardzo bogata oferta programowa sprawia, że nawet dużo lepsze komputery nie są w stanie zagrozić produktowi Commodore. Wielu obserwatorów sądziło, że możliwość pracy wprowadzonego w ubiegłym roku na rynek C128 w trybie C64 jest sygnałem o wycofywaniu się z produkcji C64, tymczasem Commodore nie zamierza z niego wcale rezygnować. Wzięto jednak pod uwagę zgłaszane często zastrzeżenia dotyczące wyglądu komputeraa. W tym roku wejdzie na rynek nowy C64 o wyglądzie niemal identycznym z C128 (niewielkie różnice o wymiarach). Wewnątrz nie będzie żadnych istotnych zmian poza mającymi na celu unowocześnienie technologii produkcji i obniżenie kosztów. Commodore poinformowała jednak, że pracuje już nad systemem operacyjnym dla C64 kompatybilnym z instalowanym obecnie, lecz umożliwiającym korzystanie z myszy i okien. (s)

Wyzwanie sztucznej inteligencji



- Rozmowa z prof. Rogerem C. Schankiem
- Inteligentny komputer
- Komputer i humanizm

Cogito ergo sum?

Z prof. Rogerem C. Schankiem, dyrektorem The Yale University Artificial Intelligence Project, psychologiem i czołowym amerykańskim specjalistą od zagadnień sztucznej inteligencji (AI) rozmawia Stewart W. Ramsey, redaktor naczelny kwartalnika Economic Impact.

Prof. R. Schank jest autorem wielu książek z tego zakresu, w tym najnowszej monografii "The Cognitive Computer: On Language, Learning and Artificial Intelligence", a także prezesem dwóch firm: Computeach Inc. i Cognitive Systems Inc., które z powodzeniem już dziś są w stanie utrzymać się ze sprzedaży produktów zaliczanych do obszaru zainteresowań sztucznej inteligencji.



Wiele ostatnio słyszy się o sztucznej inteligencji i myślących maszynach. Czy rzeczywiście maszyna może rozumować i naśladować ludzką inteligencję?

To pytanie wciąż pozostaje bez odpowiedzi. Oczywiście wielu zaprzecza takiej możliwości. Ja uważam, że nie można nazywać czegoś niemożliwym bez podjęcia próby dokonania tego.

A czy w ogóle wiemy, co to jest ludzka inteligencja?

Nie wiemy i w istocie całe przedsięwzięcie ze sztuczną inteligencją to nic innego, jak próba zrozumienia inteligencji ludzkiej.

Naukowcy mówią już o tym od lat...

Zagadnienie to jest badane od ćwierć wieku i od tego czasu nasza wiedza przeszła potężną ewolucję zarówno w zakresie rozumienia ludzi jak i maszyn. Nie można jednak mówić o żadnym przełomie, po prostu krok po kroku próbujemy robić urządzenia nieco lepsze niż budowane wcześniej. Sensacyjne doniesienia prasowe są tylko reklamą lub wynikiem dziennikarskiej ekscytacji.

Co jest główną przeszkodą w próbach nauczenia komputera myślenia symbolicznego lub posługiwania się językiem potocznym zamiast liczbami?

Pytanie jest źle postawione: komputery zawsze będą się zachowywały w ten sam sposób – bez względu na to co zostanie do nich wprowadzone, zawsze będzie następnie tłumaczone na język liczb, na język stanów "tak--nie". Pracując nad AI próbujemy zrozumieć, w jaką wiedzę trzeba komputer wyposażyć, by był on w stanie wykonać pewne zadania wymagające inteligencji. Jeśli np. maszyna ma rozumieć potoczny angielski, wówczas musimy dobrze zrozumieć, co to znaczy rozumieć język angielski. Każdy człowiek posługując się nim odwołuje się bez przerwy do swej pamięci i doświadczenia. Problem rozumienia języka to w istocie problem wyposażenia maszyny w olbrzymie doświadczenie zorganizowane w taki sposób, by możliwe było szybkie skorzystanie z tych zasobów.

Mówi się, że próbujecie dać komputerowi wiedzę o ludzkim życiu.

Oto przykład. Wybieram się w długą podróż lotniczą. W sposób naturalny musi się więc pojawić pytanie, co zrobię, gdy poczuję głód. Jeśli jednak kiedykolwiek podróżowałem samolotem, to wiem, że mogę liczyć na posiłek na pokładzie i nie muszę brać na drogę kanapek.

Nasze życia składa się z setek takich cząsteczek wiedzy o świecie, w którym żyjemy i chcąc umożliwić maszynie sensowne odpowiadania na choćby najprostsze pytanie postawione w języku naturalnym, trzeba wyposażyć ją w taką podstawową wiedzę niezbędną do interpretacji każdego zdania.

Zadanie wydaje się koszmarne: zainstalować w maszynie całą wiedzę zawartą w encyklopedii...

Jest to jeszcze trudniejsze: encyklopedia ułożona jest alfabetycznie, a wiedza ludzka w zupełnie inny sposób.

Często jedno wspomnienie kojarzy się nam z innym, choć nie ma między nimi żadnego dostrzegalnego związku. Prawdziwym problemem jest więc nie tylko zrozumienie, jaka wiedza jest nam niezbędna, ale i jak jest ona zorganizowana, w jaki sposób jedna informacja pozwala zrozumieć inną – oto dopiero przerażające zadanie.

Często w związku ze sztuczną inteligencją słyszy się o systemach ekspertalnych. Cóż to takiego?

Technicznie system ekspertalny jest to sieć reguł typu "jeśli-to", opisujących sumę pewnych doświadczeń i pozwalających komputerowi zachowywać się jak ekspert, doradzający jak postąpić w danej sytuacji. Typowym przykładem są medyczne programy diagnostyczne, pozwalające wysnuwać pewne wnioski na podstawie występujących symptomów i stanu pacjenta.

Problem z takimi systemami polega na tym, że są one zazwyczaj bardzo uproszczone, dla przykładu diagnoza opiera się na zasadach, jakich lekarze uczeni są na studiach medycznych. Rzecz jasna, prawdziwy ekspert poza wiedzą wyuczoną posiada jeszcze własne doświadczenie i każdy z nas prawdopodobnie wolałby mieć do czynienia z lekarzem, który widział już w swoim życiu wielu pacjentów z problemami podobnymi do naszych niż z młokosem, który dopiero co ukończył Akademię Medyczną. Maszyny muszą również uzyskać umiejętność korzystania z własnych doświadczeń.

Na jakie dziedziny naszego życia sztuczna inteligencja wywrze największy wpływ: politykę, edukację, medycynę czy gospodarkę?

W każdym z tych obszarów jest potencjalne zapotrzebowanie na ekspertyzy o charakterze codziennych porad. Dla przykładu: mało kto dysponuje biblioteką medyczną, tymczasem na odludnych obszarach, gdzie pomoc lekarska nie jest łatwo dostępna, możliwość uzyskania dzięki komputerowi szybkiej, elementarnej diagnozy np. choroby dziecięcej może być wielką pomocą i wręcz szansą ocalenia niejednego życia.

W przemyśle w coraz większym stopniu pewne rutynowe prace wykonywane są przez roboty. Czy robot jest formą sztucznej inteligencji, czy może on dysponować podobnie jak człowiek zmysłami: wzrokiem i dotykiem?

Robotyka do pewnego momentu była częścią sztucznej inteligencji, lecz obecnie specjaliści od robotyki większość swej uwagi koncentrują na problemach mechanicznych, np. takich jak droga, po której porusza się ramię robota.

Pewne problemy robotyki są jednak nadal ściśle związane ze sztuczną inteligencją. Należy do nich zagadnienie widzenia przez roboty. Jak robot ma rozpoznawać interesujące go przedmioty? Aby widzieć, trzeba wiedzieć, czego się spodziewać i co można zobaczyć – i w tym zakresie postęp jest wciąż niewielki. Sztuczna inteligencja musi wyposażyć robota w wiedzę, co znajduje się tam, gdzie robot usiłuje spojrzeć.

Ostatnio największe firmy samochodowe USA wykupiły część akcji pięciu spółek działających w obszarze sztucznej inteligencji. Czy oznacza to zwrot w postawie wielkiego przemysłu, na którego brak ofensywności w tym obszarze wielokrotnie Pan narzekał?

Nadal uważam, że nic nie wskazuje na poważne zainteresowanie się przemysłu sztuczną inteligencją, choć sądzę, że wspomniane przez pana firmy motoryzacyjne zrobiły dobry interes i dobrze by było, gdyby inne firmy poszły za ich przykładem. Większość przemysłu wciąż stara się uzyskać sztuczną inteligencję za darmo, bez żadnego wysiłku. Podkupują sobie specjalistów z tej dziedziny i każą im stworzyć coś niezawodnego. Wydaje się im bowiem, że sztuczna inteligencja jest znacznie bardziej zaawansowana niż w rzeczywistości.

Tymczasem dziedzina ta wymaga 5-10 lat poważnych wydatków na badania, zanim zaczną się one zwracać. Większość przedsiębiorstw przemysłowych myśli jednak z najwyżej dwuletnią perspektywą. W sposób skrajnie krótkowzroczny mówią: niech rząd sobie finansuje te badania, a my sięgniemy po ich wyniki, kiedy będą gotowe. Problem w tym, że rząd finansuje głównie badania o charakterze wojskowym, a choć w wielu wypadkach badania wojskowe doprowadziły do powstania dobrych produktów o charakterze cywilnym, to nie zawsze tak się zdarza. Ta krótkowzroczność powoduje, że Ameryka przegrywa rywalizację na dłuższy dystans.

Jaki będzie wpływ sztucznej inteligencji na rynek pracy?

Nie ma co się obawiać gwałtownych zmian. W przeszłości przeżyliśmy już wiele rewolucji przemysłowych, przeżyjemy i tę. Sztuczna inteligencja spowoduje z pewnością głębokie zmiany społeczne, będą one jednak powolne. Jak dotąd roboty są w stanie wykonywać tylko bardzo proste zadania. Uwalniają one ludzi od prac naj-

bardziej żmudnych, jak wkręcanie śrubek, lecz równocześnie umożliwiają stworzenie nowych miejsc pracy w sferze usług, a praca w usługach, w moim przekonaniu, jest postępem w porównaniu ze staniem przy linii montażowej.

Co jest głównym problemem w rozwoju sztucznej inteligencji, sprzęt czy oprogramowanie?

Sprzęt jest z roku na rok tańszy i szybszy. Naszym problemem jest oprogramowanie. Jesteśmy obecnie w samym środku rewolucji w tej dziedzinie. Przejawem jej są problemy użytkowników komputerów osobistych, kupowanych masowo bez zastanowienia, czemu właściwie mają one służyć – może grom, może redagowaniu tekstów? Poza tymi narzucającymi się zastosowaniami są jeszcze setki innych, potrzebnych prawie każdemu, dla których brak odpowiedniego oprogramowania. Jesteśmy w obliczu sytuacji podobnej do towarzyszącej niegdyś pojawieniu się telewizji. Mamy technikę, ale nie mamy programów do oglądania.

Nie jest tajemnicą międzynarodowa rywalizacja w rozwijaniu tzw. piątej generacji komputerów i superkomputerów.

Jeżeli mowa o wysiłkach Japończyków, to nie wierzę, by zdołali oni w istotny sposób zbliżyć się do celów, które, jak twierdzą, zamierzają osiągnąć.

Sądzę jednak, że generalnie rywalizacja jest rzeczą dobrą i mobilizującą. Stany Zjednoczone przez tyle lat były liderem w budowie komputerów, że – w moim przekonaniu – popadły w samozadowolenie. Wysiłki innych krajów są tu dla nas cennym dopingiem.

Jak zrozumiałem, wygląda na to, że sztuczna inteligencja jest w pewien sposób przereklamowana, a stojące przed badaczami problemy są poważniejsze, niż się na ogół przypuszcza?

To prawda. Nie twierdzę, że sztuczna inteligencja jest nierealnym mitem lub że nic z tego nie będzie, ale idzie to ciężko. Naszym głównym problemem jest nauczenie maszyny korzystania z jej własnych doświadczeń. Jak dotąd postęp jest niewielki.

Kiedy sztuczna inteligencja zdoła dowieść swego znaczenia gospodarczego, czy nie dojdzie do pogłębienia się luki technologicznej między krajami rozwiniętymi i rozwijającymi się?

Wręcz przeciwnie. Być może urządzenia o sztucznej inteligencji będą pierwszymi produktami najnowszej technologii, jakie trafią do niektórych krajów. Np. komputer służący poradą medyczną z pewnością będzie bardziej przydatny w odległej wiosce w Afryce niż w metropolii takiej jak Nowy Jork. Nawet tam gdzie nie ma elektryczności, będzie możliwe korzystanie z pracującego na bateriach mikrokomputera porozumiewającego się z wielkimi bazami danych poprzez satelitę. Komputery wcale nie są już dziś drogie i stają się coraz tańsze. Najlepszym przykładem może być kalkulator, który kosztuje dziś grosze i pozwala każdemu wykonywać skomplikowane obliczenia. Sądzę, że doradcze systemy oparte na sztucznej inteligencji upowszechnią się równie szeroko.

Opracował: W. MAJEWSKI

Inteligentny komputer?

Komputer staje się dziś niezbędny praktycznie każdemu, musi więc potrafić komunikować się ze zwykłymi ludźmi, bez specjalistycznego przygotowania. W tym celu oprócz obowiązkowej niezawodności, szybkiej pracy, dużej pojemności pamięci cechować go musi inteligencja niezbędna do tego, by rozumieć i być rozumianym. Właśnie to, a nie jakieś wymyślne zalety konstrukcyjne, stanowi istotę komputerów piątej generacji. Prace nad nimi podjęto także w ZSRR.

We wszystkich krajach rozwiniętych wyraźnie odczuwana jest dysharmonia między poziomem automatyzacji produkcji i sterowania nią. Od stu lat nieustannie rośnie udział pracowników zatrudnionych w sferze informacyjnej: nauce, oświacie, zarządzaniu i planowaniu w strukturze siły roboczej, przy równoczesnym zmniejszaniu się grupy pracowników zatrudnionych w produkcji. Przyczyna pozostaje jedna i ta sama: produkcja "automatyzuje się" bardzo aktywnie, zaś zarządzanie nią – nie. W USA inwestycje kierowane na wyposażenie robotnika w narzędzia pracy są dziesięciokrotnie większe niż nakłady na wyposażenie urzędników biurowych.

Pokonać tę barierę, lub, jak kto woli, przełamać informatyczny kryzys można jedynie pod warunkiem szerokiego zastosowania techniki obliczeniowej w planowaniu, zarządzaniu, w pracach badawczych i wdrożeniowych.

Automatyzacja pracy programistów wymaga wprowadzenia do tradycyjnej struktury komputera nowych elementów. Na rysunku przedstawiono schemat rozszerzonego komputera, potrafiącego przygotować program rozwiązania jakiegoś zadania na podstawie ustnego opisu jego warunków.

Jak widać "zwyczajny", "tradycyjny" komputer został wzbogacony o trzy nowe bloki – "procesor komunikacji", "bazę wiedzy" i "planistę". Tam właśnie syntetyzowany jest program dla "zwykłego" komputera. Wykonują więc one pracę, która wcześniej należała do programisty. Te trzy b l o k i łączy używane czasami określenie "intelektualny interfejs". "Baza wiedzy" nie jest przy tym zwykłym zbiorem faktów, formuł, twierdzeń itp. jak w obecnych prymitywnych bazach danych. Jej treść stanowią wiadomości wewnętrznie powiązane, zinterpretowane, usystematyzowane – czyli w i e d z a w pełnym tego słowa znaczeniu. Oczywiście tworzenie takich "baz wiedzy" jest zadaniem niezwykle skomplikowanym, wymagającym umiejętności uzupełniania, uogólniania i strukturyzacji przechowywanych w nich informacji, troski o to, by były one pełne i koherentne (spójne wewnętrznie, niesprzeczne). Ich tworzenie wymagać będzie rozwiązania wielu innych problemów, w wyniku których wezmą udział również psychologowie badający poznawcze struktury ludzkiej pamięci, czyli charakteryzujące ją sposoby zapamiętywania, "wywoływania" z pamięci, grupowania i systematyzowania różnych wiadomości itp.

Do zadań "procesora komunikacji" należy przełożenie pierwotnego tekstu rozwiązywanego zadania na wewnętrzny język maszyny. Ten sam blok dokonuje następnie przekładu efektów pracy maszyny na język zrozumiały dla użytkownika. "Planista" na podstawie opisu warunków zadania, otrzymanego od "procesora komunikacji", buduje roboczy program, przekazywany następnie komputerowi do wykonania. Tworzenie takiego programu odbywa się za pomocą modelu problemowej dziedziny, przechowywanej w "bazie wiedzy". Moduł ten zawiera wszystkie niezbędne wiadomości o sposobach rozwiązywania zadań z danej dziedziny. Przechowywane są w nim również standardowe programy, za pomocą których rozwiązywane są typowe zadania. Z nich, niczym z cegiełek, "planista" buduje potrzebny mu program.

Innym wyróżnikiem nadchodzącej technologii informatycznej będzie to, iż w odróżnieniu od poprzedniej praktycznie obywać się ona będzie bez papieru. Potrzebne jakiemuś użytkownikowi informacje przekazywane będą wprost na ekran monitora. Jedynie w szczególnych przypadkach niezbędne dane będą drukowane na peryferyjnej drukarce i powielane w niezbędnej liczbie egzemplarzy. Dzięki temu możliwa stanie się wreszcie oszczędność papieru, który zużywany jest obecnie bezproduktywnie w doprawdy potwornych ilościach. Jedynie amerykańskie sfery biznesu wydają na papier ok. 100 mld dol. rocznie, gdyż niesystemowe wykorzystanie techniki obliczeniowej prowadzi w istocie do dodatkowego, i to znacznego, zużycia papieru, w dodatku specjalnego, wysokiej jakości.

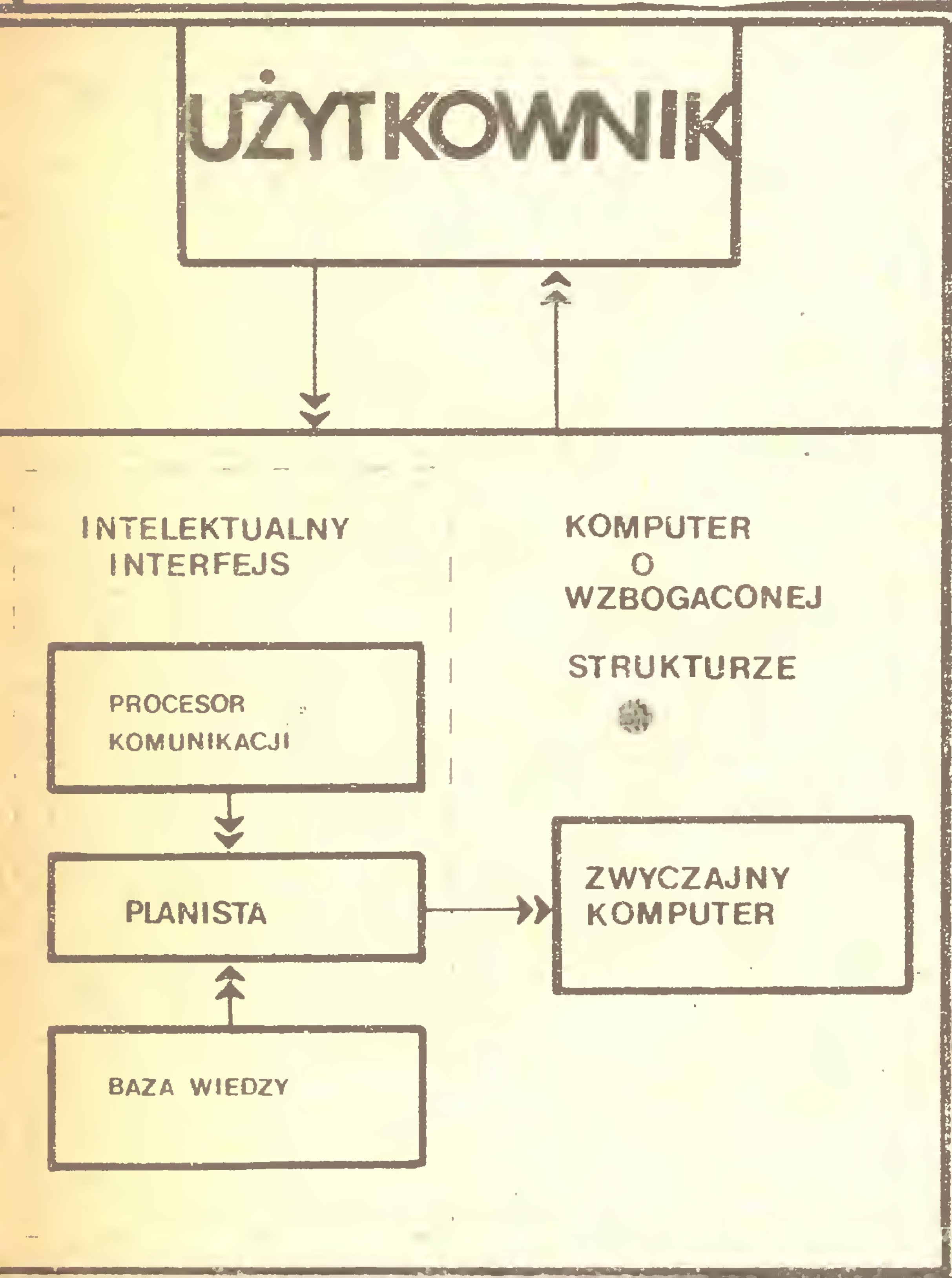
Stworzenie intelektualnego interfejsu umożliwi dostęp do komputera praktycznie każdemu człowiekowi. Kontakt z nim stanie się niewiele trudniejszy niż obsługa pralki automatycznej. Nie sądzę, byśmy pojawienia się takich urządzeń mogli oczekiwać wcześniej niż za dziesięć lat, chociaż istnieją już poszczególne elementy intelektualnego interfejsu. Zbudowano pierwszych, prymitywnych planistów, funkcjonują już systemy komunikacji umożliwiające dialog z maszyną prowadzony w naturalnym, choć znacznie ograniczonym do profesjonalnych ram, języku. Istnieją także, aczkolwiek niezbyt pojemne, bazy wiedzy. Jesteśmy świadkami narodzin nowego stylu współpracy z komputerem.

Cogito ergo sum?

Równocześnie coraz szerszym frontem wejdą w nasze życie trzy systemy nowego typu: intelektualne informacyjno-poszukiwawcze, ekspertyzowe i obliczeniowo-logiczne. Każdy z nich wart jest kilku słów opisu.

Intelektualne systemy informacyjno-poszukiwawcze są potomkami znanych od dawna "zwyczajnych" systemów informacyjno-poszukiwawczych, różniącymi się jednak od nich nie tylko nieporównanie bogatszym zapasem stale uzupełnianych wiadomości, lecz również istotną umiejętnością formułowania odpowiedzi na pytania użytkownika nawet wtedy, gdy nie mają one bezpośredniego, z punktu widzenia danego systemu, charakteru. Innymi słowy, systemy te będą na tyle "mądre", by pojąć, o co faktycznie chodzi człowiekowi, który nie potrafił dostatecznie jasno i jednoznacznie sformułować pytania. Utworzenie takich intelektualnych w pełnym tego słowa znaczeniu systemów wymaga wypracowania specjalnej pytaniowo-odpowiedziowej logiki oraz metod klasyfikacji i strukturyzacji wiedzy. Prowadzimy w tym zakresie intensywne prace.

Z trzech wymienionych najszerzej rozpowszechnione są obecnie systemy ekspertyzowe. Ich zadanie – to gromadzenie doświadczeń specjalistów, zatrudnionych w dziedzinach z trudem jedynie podlegających formalizacji – np. w medycynie, biologii, historii itp. Innymi słowy, konkretny system ekspertyzowy, zorientowany na ściśle określoną dziedzinę problemową (np. system diagnozujący ostre zapalenie nerek), jest automatycznym poradnikiem-doradcą specjalisty.



Wreszcie system obliczeniowo-logiczny jest w stanie wykonywać szereg procedur, wykorzystywanych w zadaniach związanych z projektowaniem, planowaniem itp. On właśnie odgrywać będzie decydującą rolę, albowiem istniejące metody projektowania i wdrażania nowych technologii stanowią w gruncie rzeczy przeszkodę na drodze postępu naukowo-technicznego.

W obecnym układzie wykorzystania techniki obliczeniowej w sferze badań i opracowań pomiędzy komputerem i ostatecznym jego użytkownikiem znajduje się cały łańcuszek pośredników. Użytkownik czy obecnie raczej zleceniodawca w oparciu o swą wiedzę inżynierską i doświadczenie przygotowuje, powiedzmy, wstępny projekt jakiegoś przyszłego systemu technicznego – określa składające się nań agregaty i sposoby połączenia ich w jeden system. Jego pracę kontynuuje analityk, który wykorzystując z kolei swą wiedzę z zakresu matematyki stosowanej i doskonałą znajomość przedmiotowej dziedziny, przygotowuje matematyczny model systemu i formułuje w języku matematyki zadania, jakie mają zostać wykonane za pomocą elektronicznej maszyny obliczeniowej. Z kolei pałeczkę przejmuje programista. Wykorzystując swój zasób wiedzy wnika w istotę modeli matematycznych i sformułowanych przez matematyka zadań i tym samym zmienia formułę problemu z poziomu matematycznego na programowy. Teraz programy wprowadzane są do komputera, gdzie automatycznie dokonywane jest tłumaczenie z języków wyższego rzędu na kody maszynowe. Dopiero tutaj mamy do czynienia z automatyczną translacją. Tak więc w sumie przygotowanie wariantowych obliczeń trwa bardzo długo, a reakcja całego nakreślonego systemu projektowania na ingerencje człowieka jest skrajnie powolna. A przecież ingerencje takie są nieuniknione – w każdym przypadku pojawiają się jakieś poprawki.

Dlatego też ostatecznym celem nowej technologii informatycznej jest automatyzacja przekładu z poziomu materialnego na matematyczny, a z matematycznego – na programowy.

Tworzenie nowej technologii projektowania odbywało się etapami, z których każdy stanowił jakościowy skok i sprowadzał się do kolejnego eliminowania pośredników między komputerem a jego użytkownikiem.

Pierwszy taki jakościowy skok dokonany został wraz z pojawieniem się baz wiedzy i systemów sterowania nimi. Przygotowanie programów dla komputera uległo dzięki nim gwałtownemu, blisko 100-krotnemu przyspieszeniu. A przecież pisanie takiego programu to najbardziej nużący proces. Do pracy, przez nieuwagę człowieka, nieuchronnie wkradają się błędy, co z kolei znów opóźnia przygotowanie i porównanie wariantowych obliczeń, będących wszak głównym etapem procesu projektowania. Obecnie programista ma możliwość gromadzenia biblioteki gotowych programów lub ich fragmentów – programowych modułów, z których mogą powstawać nowe programy.

Obliczenie systemów technicznych, składających się z szeregu agregatów, odbywa się obecnie w sposób następujący. W oparciu o ogólne wymagania stawiane danemu systemowi obliczenie poszczególnych agregatów powierza się komputerowi (obliczenia takie mogą być wykonywane w różnych organizacjach), a następnie wyniki zestawia się ze sobą i określa właściwości całego systemu. W przypadku, gdy jego parametry uznane zostaną za nieodpowiednie, poszczególne agregaty przeliczane są ponownie. Proces zestawiania wyników powtarza się tak długo, aż otrzymany wynik zadowoli nasze potrzeby.

Tak więc czas operacji ulega skróceniu, ale nadal jest relatywnie długi, gdyż nie zostało jeszcze zautomatyzowane zestawianie wyników poszczególnych obliczeń. Rodzi się naturalne w tej sytuacji pytanie, czy całego programu obliczeń nie dałoby się połączyć w jakiś jeden ogólny system?

W ten sposób doszliśmy do drugiego jakościowego skoku, kiedy połączenie poszczególnych programów stosowanych w jeden system prowadzi do tworzenia całych pakietów programów stosowanych. Pakiety takie bywają dwojakiego rodzaju: bądź zorientowane na metody rozwiązania, bądź na określoną dziedzinę problemową. Pakiety drugiego rodzaju – zorientowane problemowo – często wykorzystywane są w systemach projektowania i noszą wówczas nazwę systemów automatycznego projektowania SAPR (ang. CAD – Computer Aided Design). W Związku Radzieckim opracowane zostały już systemy projektowania zakładów chemicznych, samolotów itp. Wyróżniają się one szybką reakcją na ingerencję użytkownika, dzięki czemu w stosunkowo krótkim czasie można przejrzeć bardzo dużą liczbę różnych wariantów.

Wreszcie trzeci jakościowy skok w automatyzacji projektowania dokonany zostanie wówczas, gdy projektant korzystający z komputera pozbędzie się pośrednictwa programisty i wszystkich związanych z pośrednictwem tym problemów. Rozwiązanie tej kwestii zmierza w dwóch wzajemnie uzupełniających się kierunkach. Pierwszy polega na uproszczeniu i sformalizowaniu języka, jakim posługuje się człowiek, a tym samym zbliżeniu go do języka komputera. Celem jest budowa środków komunikacji, z których bez trudu nauczą się korzystać nieprofesjonalni programiści. Drugi sprowadza się do odwrotności pierwszego: język komputera zbliża się stopniowo do języka ludzkiego. Jest to w istocie bardziej gruntowne rozwiązanie problemu. Tą właśnie drogą poszli np. Japończycy opracowując ideę komputerów piątej generacji. Środki komunikacji zasługują – w przypadku drugiego kierunku – na miano prawdziwie intelektualnych, albowiem tworzone są w oparciu o programowe i techniczne środki sztucznej inteligencji. W rezultacie świat komputerów łączy się ze światem użytkownika i tym samym faktycznie przechodzimy do nowej technologii informatycznej.

Prace badawcze w zakresie intelektualnych systemów informacyjno-poszukiwawczych, a także systemów obliczeniowo-logicznych oraz ekspertyzowych osiągnęły obecnie taki poziom rozwoju, iż nadszedł czas, byśmy gruntownie zmienili tradycyjne sposoby organizacji badań naukowych i prac w zakresie projektowania. Na przykład nie ma potrzeby, by liczne instytuty naukowo-badawcze brały udział wraz z biurami konstrukcyjnymi w opracowaniu konkretnych systemów stosowanych. Ich głównym zadaniem zaczyna być obecnie prowadzenie badań, wzbogacających bazy wiedzy systemów nowego typu, z których korzystać będą konstruktorzy i projektanci. W ten sposób dojdzie do dalszego podziału pracy w sferze tworzenia nowych maszyn, urządzeń i materiałów, powstaną warunki dla jej dalszej specjalizacji.

Tak więc związek pomiędzy teorią sztucznej inteligencji, techniką obliczeniową i przemysłową robotyką jest na tyle ścisły, iż obecnie żaden z tych kierunków nie może rozwijać się bez uwzględniania interesów i celów pozostałych. W ciągu najbliższego dziesięciolecia przejście do technologii, przy której użytkownik, bez pośrednictwa programisty, będzie bezpośrednio pracować na komputerze, stanie się powszechne. W ten sposób połączone zostaną w jedną całość badania czysto naukowe ze stosowanymi i wręcz czysto technicznymi.

opr. MAREK CAR
na podstawie "Znanije-Siła" nr 2/86

Autor – Hermogen S. Pospiełow, członek AN ZSRR – jest przewodniczącym Rady Naukowej programu badań w zakresie sztucznej inteligencji.

Komputer i humanizm

Z prof. MIECZYSŁAWEM GOGACZEM, kierownikiem Katedry Historii Filozofii Akademii Teologii Katolickiej w Warszawie rozmawiają Stanisław Krajski i Krzysztof Letki

Czy komputeryzacja zmienia kulturę?

Skonstruowanie komputera jest wydarzeniem porównywalnym z wynalezieniem pisma i druku: powstał nowy, dostępniejszy sposób utrwalania i przekazywania kultury, nowe narzędzie jej rozwoju, umożliwiające szybszą wymianę doświadczeń i osiągnięć ludzkich. Kultura może stać się powszechniejsza i w tym sensie komputer sprzyja duchowemu rozwojowi człowieka, tworząc szansę wyrównania startu kulturowego wszystkich ludzi, ułatwienia go.

Przez kulturę rozumiem przy tym usprawnienia intelektualne i wolitywne oraz utrwalenie przemyśleń i decyzji w dziełach zewnętrznych, ułatwiających i przyspieszających usprawnienia wewnętrzne (określenie to jest zrozumiałe dla filozofów, lecz podstawy języka filozofii powinien znać zarówno każdy humanista, jak inżynier).

Czy oprócz nadziei komputeryzacja niesie nam jakieś zagrożenia?

Niebezpieczeństwa są zawsze te same. Ukazuje je historia kultury, a zwłaszcza filozofii. Stałym zagrożeniem jest przypisywanie temu, co pomyślane lub wytworzone, pozycji realnego bytu. Przypisuje się więc komputerom inteligencję, niezależność od człowieka, a czasem nawet wyższość nad człowiekiem. Tymczasem komputer, choć pewne działania może wykonywać sprawniej niż człowiek, zawsze pozostanie wytworem człowieka, przez niego zaprogramowanym i wypełnionym pomysłami. Kwestią etyki i sztuki programującego jest niedopuszczenie, by komputer, samodzielnie kojarząc różne kody, podjął atak na człowieka lub stał się narzędziem takiego ataku. Rozważany w literaturze SF tzw. bunt robotów nie może mieć źródła w samodzielnym pomyśle komputera, lecz w niedoskonałości programowania go przez człowieka, na którego spada obowiązek myślenia na tyle dalekowzrocznego i konsekwentnego, by nie doszło do ziszczenia się baśni o uczniu czarnoksiężnika.

Nasza uwaga musi się więc przenieść z komputera na człowieka, który go stworzył i który go obsługuje...

Człowiek z racji swej duchowo-fizycznej budowy nie wszystko może zrealizować. Dzięki jednak intelektowi o wiele więcej może poznać niż wykonać. Nie może np. bezpośrednio wykonywać prac w wysokich temperaturach, w materiałach promieniotwórczych, nawet w szybki sposób przenosić się z miejsca na miejsce. Jest jednak w stanie obmyślić maszyny, które go zastępują w wykonywaniu tych działań, a nawet więcej: precyzyjniej niż człowiek wykonują skomplikowane działania techniczne. Człowiekowi pozostaje zatem przede wszystkim humanistyka, która wypełniając nasze życie osobowe, zarazem przygotowuje inżynierów do projektowania techniki chroniącej człowieka. Dodajmy, że technika, która nie jest wierna człowiekowi, jest antyhumanistyczna i do niczego nie jest nam potrzebna. Wysokiemu rozwojowi techniki musi więc towarzyszyć troska o wysoki poziom humanistyki, w tym filozofii rzeczywistości, filozofii człowieka, etyki i teorii społeczeństwa. Te nauki rozwijają i pogłębiają myślenie, które jest wciąż napędem pomysłowości technicznej. Technika jest bowiem zawsze na miarę myślenia człowieka.

A co Pan Profesor sądzi o tworzeniu kultury przez komputery? Malują one przecież obrazy, komponują muzykę i teksty poetyckie. Czy komputery mogą zastąpić człowieka w tworzeniu kultury lub choćby niektórych jej dziedzin? Jaką wartość mają, zdaniem Pana Profesora, kulturowe wytwory komputerów?

Zachowanie komputerów musi być zgodne z ich naturą. Możliwe jest zatem łączenie przez nie dźwięków, plam czy nawet słów w ich warstwie znaku fizycznego. Wydaje się, że komputer mógłby odkryć poezję dadaistyczną. Nawet więcej. Komputer może nas zastąpić w działaniach podmiotowanych przez zmysłowe władze poznawcze. Jego natura nigdy jednak nie umożliwi mu zastąpienia nas w działalności intelektualnej w zakresie relacji istnieniowych i osobowych. Komputer nie może zaproponować etyki, metafizyki, nie uzyska świadomości chronienia relacji osobowych i samych osób. Komputery wymagają takiej opieki, jaką stosujemy wobec małych dzieci i zwierząt. Trzeba je nauczyć prawidłowych odniesień i nieustannie kontrolować. Komputery nie mając własnego myślenia nigdy nie zastąpią człowieka w jego pełnym życiu umysłowym. Mogą jednak zachowywać się jak źle wytresowane zwierzęta operujące tylko zdolnościami zmysłowymi. Jest to oczywiście dość groźna perspektywa. Jest ona jednak do opanowania, gdy twórcy komputerów i ci, którzy je obsługują, będą głęboko ugruntowani w humanizmie. Wartość tego, co tworzą komputery, ma więc poziom tylko fizycznej i zmysłowej aktywności, w wypadku bytów żywych opartej na instynkcie, w wypadku komputerów na zakodowaniu. Możemy to ująć prościej: samochód jadący bez kierowcy będzie zabijał przechodniów i sam rozbije się na napotkanej przeszkodzie. Wynika z tego apel do techników o kierowanie się rozsądkiem, miłością do ludzi, o postawę służenia dobru, o głębokie wykształcenie humanistyczne i oczywiście techniczne.

Im wyższa więc komputeryzacja, tym lepsze powinno być nasze wykształcenie humanistyczne. Jak winno ono wyglądać?

W każdej dziedzinie ważne są umiejętności, nade wszystko jednak liczy się to, jakim się jest człowiekiem. Tymczasem we współczesnym kształceniu stawia się głównie na umiejętności, a niekiedy tylko na wiedzę: jest ono zorientowane na użyteczność. Tymczasem użyteczność może dotyczyć wytworów, a nie człowieka. Wychowanie pozostawia się więc rodzinie i strukturom pozaszkolnym.

Człowieka na drugim planie usytuowano nie tylko w szkolnictwie, ale i w całej współczesnej kulturze. Pierwsze miejsce zajmuje postulat nowości i postępu, a więc pierwszeństwo rzeczy przed człowiekiem. Ta dysproporcja tylko pozornie służy rozwojowi techniki, gdyż bez pełnego myślenia człowieka nie może się ona rozwijać. Jesteśmy więc w okresie programów krótkowzrocznych, obok takich wytworów jak broń masowej zagłady zagrażają nam braki w dziedzinie humanistycznego i moralnego wychowania ludzi. To wychowanie wymaga kierowania do bogatszego niż techniczne zespołu wartości.

Jawi się tu problem duchowego wymiaru człowieka i problem Boga. W środowiskach technicznych modne jest pomijanie tego aspektu. W związku z dziewiętnastowiecznym modelem uczonego prawie nas peszy wierzący inżynier. Tymczasem wielki rozwój wysokiej technologii dopomina się o wysokie, prawie ostateczne kryteria postępowania człowieka. Trzeba więc zadbać o przyzwyczajenie do tych kryteriów. Wprost trzeba się ich nauczyć. Prawdy i dobra nie można prawidłowo określić bez identyfikacji rzeczywistości, czyli bez filozofii bytu. Jest to program normalnej humanistyki uniwersyteckiej i chyba błędem było wydzielenie szkół technicznych z uniwersytetów. Skazuje to je bowiem na zawężenie perspektywy i skupienie uwagi tylko na wytworach. Żadne wytwory nie są same z siebie szkodliwe. Błędne i szkodliwe może być tylko ich zaprogramowanie. Technika ujawnia poziom humanistyczny i moralny człowieka. I raz jeszcze powtórzmy – technika funkcjonując w kulturze domaga się od nas wysokich norm służenia człowiekowi. Konieczna jest też odpowiedzialność, wzięcie na siebie skutków naszych działań. Wymaga ona konsekwentnego myślenia i przewidywania tych skutków, które może przewidywać i odkrywać tylko człowiek. Nigdy nie określi ich i nie oceni komputer. Kierowanie się w działaniach prawdą i dobrem sytuuje nas w mądrości. Mądrość bowiem jest taką umiejętnością intelektu, dzięki której ujmuje on wiedzę w odniesieniu do dobra każdego bytu. Możemy więc powiedzieć, że technika i człowiek muszą kierować się mądrością.

Czym w takim razie jest według Pana Profesora komputer?

Jest to wytwór człowieka i jego istota wypełnia się w osiągnięciu zamierzonego przez człowieka celu. Ten cel zawsze jest kulturowy i na miarę kultury, człowieka i czasów.

Kino i komputery

Na razie więc nie komputer myśli jak człowiek, ale programistów uczy się "myśleć jak komputer", to znaczy krok po kroku, niczego nie omijając. Trzeba zapomnieć o intuicji, interpolacje dokładnie opisywać, przy ekstrapolacjach się zastrzegać i opisywać je równie mozolnie... Każda rzecz, z którą ma się w przyszłości uporać komputer, musi być dokładnie znana i opisana. Jeśli pojawi się problem, którego nie możemy zawczasu opisać, to wprowadza się zazwyczaj wiele warunkowych rozwiązań. Co więcej, trzeba te warunkowe rozwiązania mieć cały czas w pamięci i nie zapominać, że jest coraz więcej dróg dotarcia przez komputer do jednego z celów, na których nam zależy. Nieuniknienie pojawia się wtedy jakiś błąd w programie, który da się wykryć tylko przy testowaniu "jak w życiu", o tych bowiem, które ujawniają się w zwyczajnym przeglądaniu programu po jego napisaniu, w ogóle nie warto wspominać. "Myślenie jak komputer" oznacza wszak działanie na jednej rzeczy w danym momencie, nieuchronnie zatem musimy dojść do punktu, w którym zbyt wiele rozwiązań jest warunkowych, zależnych od nieznanych czynników. Gdy zaś w jakimś systemie wiele komputerów pracuje nad jedną rzeczą na raz (ale każdy sobie), rozwiązywane przez nie problemy zostają doprowadzone do takiego miejsca, w którym większość z nich zaczyna zależeć od tego, co – także warunkowo – działo się w innych komputerach. Program przestaje być sekwencyjny – z punktu widzenia jego analizy nie można już mówić o "myśleniu jak komputer". O jakości takiego programu można wyrokować jedynie po długotrwałych i mozolnych próbach.

Wynika z tego, że napisanie i zrozumienie bardzo dużych programów na bieżąco przetwarzających dane – leży poza polem obecnych możliwości intelektualnych człowieka. Program taki można sporządzić opracowując jego szkic a następnie uzupełniając go, metodą prób i błędów, w toku kolejnych uruchomień. Jest to jednak niezwykle czasochłonne.

Przy oprogramowaniu handlowym wyszukiwanie błędów przerywa się nie wtedy, gdy już ich nie ma, ale gdy firma uznaje, że następne dałoby się wprawdzie wykryć, ale przy zbyt wysokich kosztach. Użytkownicy wiedzą zaś, że nie istnieje oprogramowanie wolne od wad i godzą się na taką sytuację.

Nie ma znanej obecnie techniki programowania, która by usuwała te strukturalne ograniczenia.

NOWE JĘZYKI, PROGRAMOWANIE AUTOMATYCZNE

Pierwsza praca o automatycznym programowaniu powstała w latach czterdziestych (napisał ją Saul Gorn). Przez jakiś czas była utajniona. Na tytułowe pytanie "Czy jest możliwe automatyczne programowanie?" odpowiedź dawała twierdzącą. Ale w owych czasach programiści osobiście dziurkowali każdą taśmę i to, o czym pisał Gorn – wedle dzisiejszej terminologii jest to po prostu asembler, translator języka symbolicznego. Potem, w miarę pojawiania się języków coraz wyższego rzędu, poprzeczka "automatycznego programowania" podnosiła się i nazywano tak już efekty stosowania języków takich jak Fortran czy Algol.

Termin "programowanie automatyczne" zawsze był eufemizmem określającym język programowania wyższy niż dostępny programistom na danym etapie historycznym.

Stosowanie lepszych języków prowadzi do zmniejszenia liczby szczegółów, z którymi ma do czynienia programista, dzięki czemu zwiększa się niezawodność oprogramowania. Ale nie można oczekiwać tu całkowitego zastąpienia człowieka przez maszynę. Nie należy także oczekiwać szybkiego pojawienia się języków jeszcze doskonalszych. Można przewidywać pewne uproszczenie obecnie znanych języków kompilujących, ale nie rewolucję. Wielu ekspertów uważa znane i stosowane obecnie języki za wystarczająco dobre. Zastrzeżenia podnosi się nie wobec nich, ale wobec zadań, jakie mają wypełnić przy "wojnach gwiezdnych".

CECHY PROGRAMU "WOJEN GWIEZDNYCH"

System komputerowy miałby, przy pomocy ogromnej liczby bardzo różnie działających czujników, zidentyfikować i ustalić położenie celów, a następnie skierować na nie broń. Charakterystyki celów nie mogą być dokładnie poznane przed momentem rozpoczęcia się bitwy – to wynika z elementarnych praw wojny.

Funkcjonowanie całego systemu czujników, broni i układów naprowadzających będzie zakłócane przez nieprzyjaciela przy pomocy nieznanych metod; podobnie system dublujący stanie przed identycznym problemem. Nieznana jest zatem liczba elementów całego systemu "gwiezdnych wojen".

Czas reakcji – jak już pisałem – będzie tak krótki, że mowy być nie może o interwencji ludzkiej, nie wspominając już o ewentualnych korektach programu w przypadku stwierdzenia nieprawidłowości lub oczywistych błędów. Konieczny czas reakcji określa zarazem nieprzekraczalny termin ukończenia obliczeń przez komputery, a tego elementu także nie da się określić ze względu na trudne do przewidzenia charakterystyki poruszania się celów. Nie można nawet określić "sytuacji najmniej korzystnej".

Każdy z elementów systemu, układu czujników i układu broni będzie dysponował własnym oprogramowaniem, poza "ogólnym", rządzącym całością. Układy te będą musiały być wzbogacane jakościowo i ilościowo jeszcze w fazie projektowania, nie mówiąc już o doskonaleniu po zainstalowaniu. Cechy broni i czujników także nie są jeszcze znane...

Pisząc programy sterujące układami broni należy przyjąć jakieś założenia co do celów, które mają one zwalczać. Jeśli ich charakterystyki istotnie się zmienią – oprogramowanie nie będzie przystosowane i początkowo niewielki błąd w naprowadzeniu broni może stopniowo narastać, aż do bezużyteczności broni. Wynika stąd potrzeba ciągłego testowania programów "wojen gwiezdnych". Najbardziej złożone programy – jak pokazuje historia rozwoju technik programowania – powstawały jako przynajmniej częściowa kontynuacja poprzednich programów o podobnym charakterze.

Ale tutaj nie będzie możliwości przeprowadzenia testu "jak w życiu"...

GRANICE REALIZMU

Koncepcja "wojen gwiezdnych" jest dokładnie opisana w tzw. raporcie Fletchera sporządzonym w 1983 r., którego pełna nazwa brzmi: "Raport z badań nad wyeliminowaniem zagrożenia ze strony międzykontynentalnych pocisków jądrowych"; wydawcą jest Departament Obrony USA. Zawiera on wiele rozdziałów, w których przedyskutowano wszystkie elementy systemu – podsystemu broni, czujników, układów naprowadzania itp. Utajniona jest część omawiająca strategiczne założenia wojny w kosmosie, konstrukcja systemu; pod suknem schowano nawet rozważania o broniach, których jeszcze nie ma. Tajna jest całość – z jednym znamiennym wyjątkiem. Każdy może się zapoznać z rozdziałem o oprogramowaniu. Należy z tego wnosić, że prace są tak daleko w lesie, iż nie ma jeszcze o czym mówić serio. Z powodów, które nie są wymienione w raporcie Fletchera, ale o których napisałem.

Aby cały utopijny system "wojen gwiezdnych" mógł powstać, w dziedzinie oprogramowania musiałby się ujawnić jakiś nadzwyczajny duch odkrywczy wielkich zespołów badawczych pracujących w pocie czoła. Dlatego rząd amerykański tak usilnie zabiega o wciągnięcie wszystkich krajów kapitalistycznych do prac badawczych. Ale nawet przy wykorzystaniu ich potencjałów, działanie tego "systemu obronnego" jest bardzo niepewne.

Dlaczego zatem USA tak się angażują w "wojny gwiezdne"? Są opinie, że wynika to z pomylenia istoty barier technicznych i barier intelektualnych. Pierwsze z nich stosunkowo łatwo pokonać: to na przykład problem typu więcej, szybciej, wyżej, dalej, silniej, taniej itp. Wyniki są tu prostą funkcją liczby osób pracujących nad jakimś zagadnieniem i funduszy, którymi dysponują.

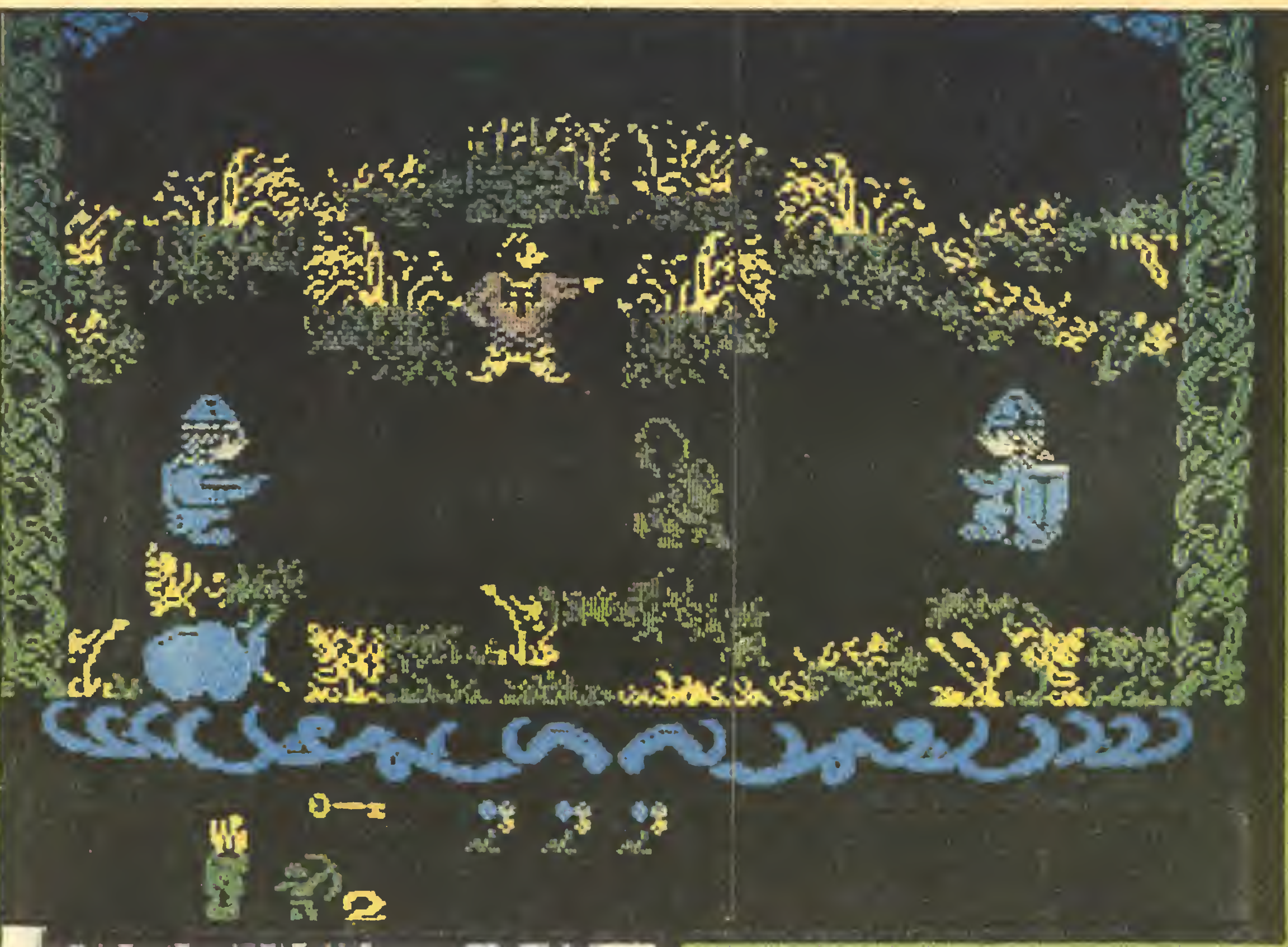
Drugi rodzaj barier nie jest tak oczywisty, nie każdy dostrzega nawet ich istnienie. Wizjonerzy i pisarze SF od lat opisują interesujący nas stan docelowy – tzn. sytuacje, kiedy komputery mogą dosłownie wszystko. Błędy w rozumieniu tej bariery przejawiają się na przykład w mniemaniu, że obecne komputery liczą zbyt wolno jak na potrzeby "wojen gwiezdnych". Rzeczywiście, elektronika świetlna lub technologia bazująca na arsenku galu, umożliwi przyspieszenie obliczania – tylko po co? Nie może bowiem powstać przy obecnym stanie sztuki programowania system, w którym ową ogromną szybkość można by spożytkować w "wojnach gwiezdnych". Widzimy tu, świadome lub nie, zastępowanie w wyobraźni barier nieprzekraczalnych przez bariery dające się pokonać. Następuje samouspokojenie i próby wdrożenia tego, o czym traktuje literatura SF... Tak bowiem należy oceniać politykę rządu USA w omawianej dziedzinie.

Na każdym programie zakupywanym w sklepach amerykańskich można przeczytać taką oto formułkę "umycia rąk": "Nie udzielamy gwarancji ani też nie wypowiadamy się otwarcie czy w sposób dorozumiany o oprogramowaniu opisanym w niniejszym podręczniku – ani co do jego jakości, wartości handlowej, ani co do zdolności przetwarzania danych czy przystosowania do jakiegoś zadania . Oprogramowanie to jest zatem sprzedawane "jak jest" i nabywca-użytkownik przyjmuje na siebie całkowite ryzyko i odpowiedzialność wynikającą z funkcjonowania programu. W żadnym zaś wypadku wydawca programu nie może być pociągnięty do odpowiedzialności za bezpośrednie, pośrednie, charakterystyczne dla tego programu lub zgoła wyjątkowe szkody powstałe w wyniku błędu tkwiącego w nim, nawet jeśli wydawca został powiadomiony o możliwości powstania takich szkód"...

JACEK A. LIKOWSKI

Ps. Wielu szarych ludzi jest przekonanych, że "wojny gwiezdne" można wygrać, bo przecież na własne oczy widziało w telewizji działające systemy broni niszczące każdą rakietę... Jest jednak różnica między filmem animowanym a prawdziwym polem bitwy. Poza tym oprogramować maszynkę do robienia kreskówek już ludzkość potrafi.

Rozkosze łamania palców



Historia Robin Hooda odżywa na nowo w zręcznościowo-przygodowej grze "ROBIN OF THE WOOD", opublikowanej przez firmę ODIN COMPUTER GRAPHICS.

Legendy tamtych czasów funkcjonowały głównie w formie ustnych przekazów, stąd różnorodność wersji i pewna dowolność w traktowaniu faktów. Ta opowiada o Robinie, synu Alerica. Aleric był strażnikiem Srebrnej Strzały, która stanowiła dla narodu Saksonów symbol wolności i pokoju. Niestety ziemie Saksonów najechali Normanowie, niszcząc i zabijając. Zarządca okręgu Nottingham uśmiercił Alerica i przywłaszczył sobie Srebrną Strzałę. Nastały ciężkie czasy dla Saksonów, pełne gwałtu i grabieży.

Gdy Robin dorósł do noszenia broni, stał się dla Normanów groźnym przeciwnikiem. Odbierał najeźdźcom zrabowane mienie i rozdawał biednym. Wychowany w lesie, bez trudu unikał pogoni. Szeryf z Nottingham, znając znaczenie Srebrnej Strzały i wiedząc, że Robin nie oprze się wyzwaniu, postanowił podstępnie zwabić go do miasta. Ogłosił więc konkurs strzelania z łuku, w którym główną nagrodą będzie ta właśnie Srebrna Strzała.



W tym miejscu legendy rozpoczyna się Twoja misja. Lecz zanim staniesz w szranki z innymi łucznikami, musisz wykonać wcześniej kilka innych, równie trudnych i niebezpiecznych zadań. By przystąpić do turnieju i uniknąć natychmiastowego rozpoznania, musisz mieć trzy magiczne strzały. Są one w posiadaniu starego, mądrego Enta (Enty – to Ludzie-Drzewa, pamiętamy je z "Władcy Pierścieni" Tolkiena). Ent ma ponadto Twój łuk oraz miecz – żeby je otrzymać, musisz dać w zamian po trzy sakiewki złota za każdą sztukę broni. Zdobycie złota będzie jednak wymagało walki z Normanami, stanowiącymi eskortę złego i bogatego Biskupa z Peterborough, którego trzeba ograbić (okazuje się, że egalitaryzm wynaleziono dużo wcześniej i gdzie indziej).



Teren gry podzielony jest na trzy części. Największa z nich to obszary leśne, następna to zamek, w którym odbędzie się turniej oraz lochy zamczyska, do których często będziemy wrzucani. Las stanowi skomplikowany labirynt zarośli i drzew (tylko dlaczego drzewa są niebiesko-czerwone?). Po lesie biegają czerwone dziki, których trzeba unikać, gdyż każde spotkanie z nimi kończy się poważnym zranieniem. Robin rozpoczyna grę uzbrojony tylko w krótką pałkę. Wystarczy ona jednak na pokonanie włóczących się wszędzie wojowników normańskich, którzy mają rozkaz zabić Robina i strzelają bez ostrzeżenia. W lesie można też spotkać zbierającą zioła czarownicę, która będzie żądać od Ciebie daniny, a jeśli jej oczekiwań nie spełnisz, wtrąci Cię do lochów (żeby wydostać się z nich, należy wcześniej znaleźć klucz). Jeżeli jednak będziesz miał już zebraną wystarczającą liczbę ziół (wyglądają jak kwiatki), czarownica przeniesie Cię w nagrodę w inny obszar lasu, oszczędzając Ci trudów marszu. Niestety zabiera również z trudem zebrane sakiewki złota, a przenosi zazwyczaj w niezbyt odległe miejsce – stąd czarownicy, jak w życiu, należy unikać, mimo że wygląda sympatycznie. Zioła, kołczany pełne strzał, a także premię w postaci dodatkowego "życia" można znaleźć w poszyciu leśnym.

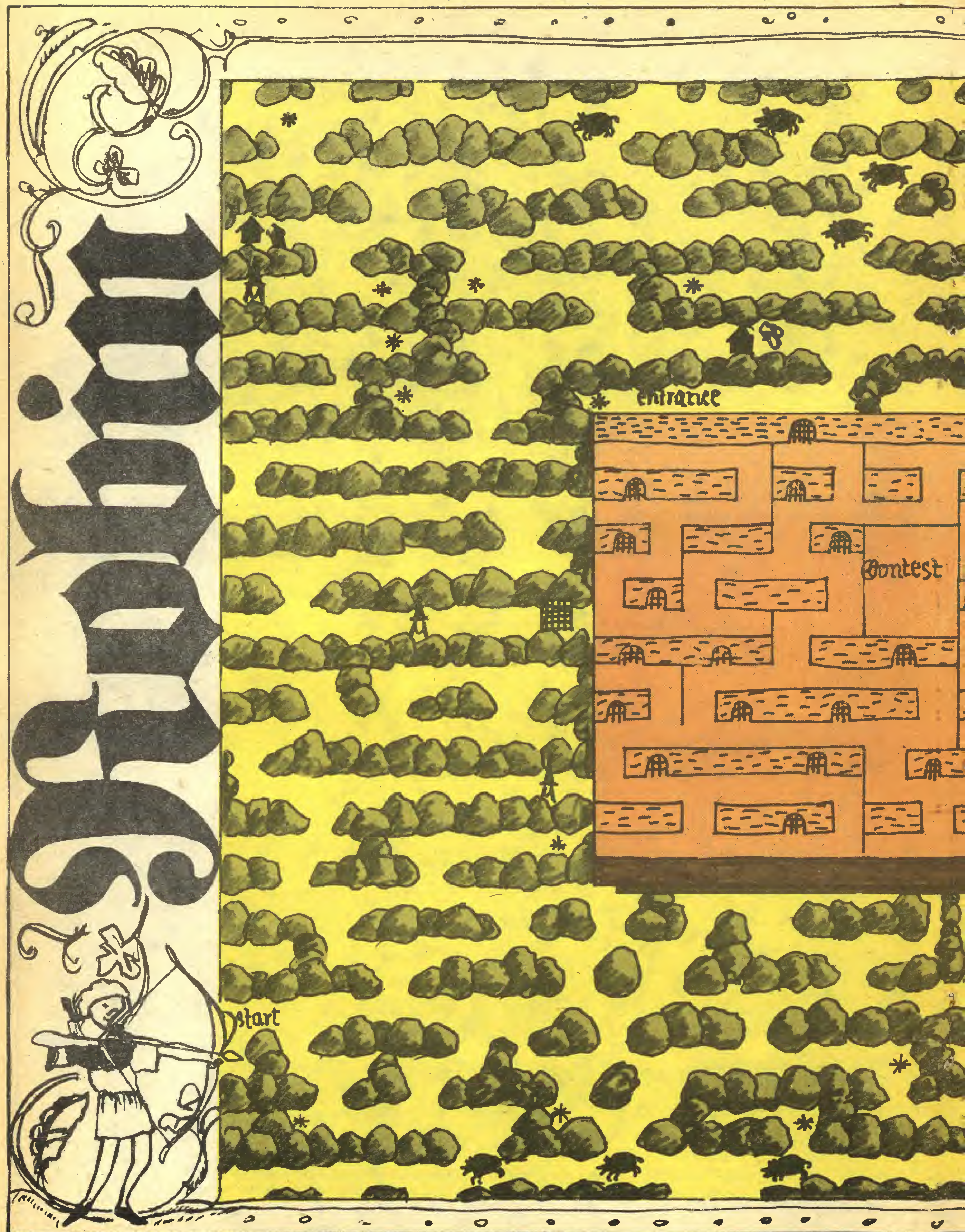
Zebrane i posiadane przedmioty wyświetlane są w dolnej części ekranu. Powyżej znajduje się skomplikowany ornament (stylizowane rogi jelenia), którego kolor symbolizuje stan zdrowia Robina. Każda rana otrzymana w walce obniża jego siły i jeżeli nie będzie wystarczająco zręczny i szybki, umrze, nie wykonując zadania. Bardzo pomocne może być odwiedzenie starego pustelnika, który leczy rany. Pamiętać jednak należy, że pustelnik nie znosi broni w pobliżu swojej chatki i może wówczas być bardzo nieprzyjemny.


Ponadto po lesie wędruje Biskup z Peterborough, chroniony przez normańskich wojowników. Wystarczy pozbawić go eskorty i Biskup porzuca pełne złota sakiewki, salwując się ucieczką. Czasem w swoich wędrówkach Robin może spotkać samego Szeryfa z Nottingham, co zawsze kończy się wtrąceniem do lochów.

Zadanie, jakie Robin ma przed sobą, jest więc równie trudne jak odpowiedzialne i Ty musisz mu w nim pomóc. Poznanie i zapamiętanie 330 lokacji wymaga poświęcenia kilku godzin na próby, bardzo przy tym pomocne jest wykonanie mapy terenu.

Gra jest ciekawa graficznie, choć dobór kolorów może budzić zdziwienie. Dźwięk, jak na możliwości Spectrum, dobrze odzwierciedla sytuację na ekranie, interesująca jest zwłaszcza synteza mowy i melodyjka na wstępie. Program daje pełny wybór sterowania, w tym możliwość zdefiniowania klawiszy.

GRZEGORZ CZAPKIEWICZ


entrance
Contest
start



start
entrance
start
teleport point
start
S. Szczypka '86 based on "CRASH"

Rozkosze łamania palców

Poke n, ∞

Szersze omówienie programu "ROBIN OF THE WOOD" firmy ODIN znajdziecie obok – tutaj tylko poprawki ułatwiające (powiedziałbym nawet: umożliwiające) skończenie tej gry. Wprowadzamy je tak, jak omawiane w poprzednich numerach – przy wykorzystaniu programu "COPY COPY". Pierwsze dwa segmenty programu kopiujemy bez zmian, a następnie wgrywamy trzeci od adresu 23424 (przy pomocy rozkazu LOAD AT 23424). Ten segment jest bez nagłówka i ma długość 42112 bajtów.

Popularne "wieczne życie" uzyskujemy wpisując POKE 48690,0. Jednym z utrudnień gry jest czarownica, pojawiająca się nieoczekiwanie i zabierająca nam z trudem zebrane przedmioty. POKE 55857,0 usuwa ją całkowicie, ale niestety ta poprawka uniemożliwia ukończenie gry. Może jednak posłużyć do wykonania mapy lub jako wersja treningowa. Warunkiem koniecznym zakończenia gry jest dotarcie do zamku, a jedyna droga, jaka tam prowadzi, to teleportacja. I właśnie czarownica nas przenosi, odbierając w zamian trzy kwiatki. Żeby jednak czarownica nie odbierała nam również sakiewek ze złotem (tego przecież nikt nie lubi), wpisujemy 0 w kolejne trzy adresy: POKE 56154,0, POKE 56155,0 i POKE 56156,0. Po wprowadzeniu powyższej zmiany musimy w grze wystrzegać się zebrania trzech kwiatków, zanim wykupimy od starego Enta wszystkie potrzebne do jej ukończenia przedmioty (miecz, łuk i trzy magiczne strzały). Gra, początkowo interesująca, wkrótce staje się nudna, gdyż zebranie co najmniej piętnastu sakiewek złota wymaga wielokrotnego przejścia labiryntu. Możemy jednak oszukać starego Enta wpisując POKE 57154,0, POKE 57155,0, POKE 57156,0 oraz POKE 57171,0 i POKE 57172,0. Nie zauważy on wówczas, że nie odebrał należnej zapłaty. Na zakończenie zmiana dla tych, którzy nie lubią oddawać nawet kwiatków. POKE 56106,6, POKE 56107,0, POKE 56108,0 i czarownica dokonuje teleportacji nie biorąc nic w zamian. W tym przypadku konieczne jest również przestrzeganie zasady omijania kwiatków przed wykupieniem uzbrojenia.

Miejsce, w którym możemy znaleźć Enta, zmienia się losowo w każdej grze. Niekiedy jest bardzo niekorzystne, uniemożliwiające właściwie skończenie gry. W normalnej wersji nie stwarza to większego problemu, gdyż i tak po chwili wracamy do początku gry. Po wprowadzeniu ułatwiających życie poprawek jedynym rozwiązaniem jest naciśnięcie CAPS SHIFT i BREAK.

Po wprowadzeniu wybranych poprawek nagrywamy segment na taśmę, dołączając go do dwóch nagranych już wcześniej.

Powyższe poprawki dotyczą wersji "ROBIN OF THE WOOD", przeznaczonej dla ZX Spectrum. Dzięki pomocy Tomasza Pycia i w oparciu o brytyjskie pisma AMSTRAD ACTION oraz POPULAR COMPUTING WEEKLY przedstawię teraz krótkie programy, pozwalające użytkownikom Amstrada wprowadzić zmiany, dające "wieczne życie" w kilku grach. Przed wpisaniem tych poprawek należy znaleźć na taśmie miejsce, w którym zaczyna się interesujący nas program, a następnie wyczyścić pamięć maszyny przez naciśnięcie: CTRL+SHIFT +ESC.

Przedstawiane gry są typowo zręcznościowe i nie wymagają szerszego opisu. Pierwsza z nich to "PUNCHY".

Wpisujemy program (Listing 1) i uruchamiamy go przez wciśnięcie CTRL+"mały" ENTER, a następnie PLAY w magnetofonie i dowolny klawisz, tak jak przy innych programach.

1

```
10 MEMORY &1FFF
20 LOAD "CODE"
30 POKE &20A9,255
40 CALL &2000
```

W "ROLAND ON THE ROPES" będziemy śmielej się wspinać, jeżeli wpiszemy: (Listing 2)

2

```
10 MEMORY 4800
20 LOAD"ROLAND.D"
30 LOAD"ROLAND.C"
35 INK 0,1 : INK 1,24 : INK 2,20 : INK 3,6
40 POKE -25804,0 : POKE -25562,0
50 CALL 41100
```

W linii 40 muszą być minusy. Teraz bierzemy w rękę młotek i poprawiamy "KILLER GORILLA": (Listing 3)

3

```
10 MODE 1
20 MEMORY &3FFF
30 WINDOW 8,33,18,20
40 INK 0,0 : BORDER 0 : INK 2,6 : INK 3,26
45 PRINT "KILLER GORILLA is loading" : PRINT :
    PRINT "Please wait..."
50 LOAD "!SCREEN"
60 CALL &8200 : MEMORY &1FFF
70 LOAD "!GAME"
80 POKE 17429,0
90 CALL &4100
```

Linię 45 możemy pominąć (naśladuje oryginalny loader). Postępując podobnie jak poprzednio, tzn. czyszcząc pamięć, wpisując program i startując magnetofon otrzymujemy wersję z "wiecznym życiem".

Powyższe gry nie stanowią z pewnością awangardy programów dostępnych dla użytkowników Amstradów. Obecnie najwyższe oceny (wg pism angielskich) uzyskują : "SPINDIZZY" firmy Electric Dreams, "HITCHHIKER'S GUIDE TO GALAXY" wydane przez Infocom oraz "CRAFTON & XUNK" – firmy PSS. Nie dotarły jeszcze amstradowskie wersje "LORD OF THE RINGS", "HIGHWAY ENCOUNTER" czy "DALEY THOMPSON'S SUPERTEST", znane już posiadaczom Spectrum, którzy dla odmiany nie widzieli jeszcze "SPELLBOUND". Myślę jednak, że w najbliższym czasie wyrównają się różnice ilościowe i wśród programów przeznaczonych na Amstrada zacznie zwyciężać jakość.

GRZEGORZ CZAPKIEWICZ

Tomahawk



DI – Digital Integration – to nazwa firmy znana każdemu entuzjaście gier symulacyjnych na ZX Spectrum. DI wprowadziła na rynek w 1983 roku program FIGHTER PILOT – uznany za najlepszy program symulacyjny dostępny na Spectrum.

Od tej chwili Dave Marshall i Rod Swift (założyciele DI) pracowali nad programem TOMAHAWK, symulatorem lotu i walki helikoptera US Air Force – Apacz. Program ten wszedł na rynek pod koniec 1985 roku i ma wszelkie szanse stać się kolejnym przebojem.

Pierwszą rzeczą rzucającą się w oczy jest duża liczba znajdujących się na ziemi obiektów: drzew, gór, dział artylerii p/lotniczej, czołgów i różnych budowli. Wszystko narysowane jest co prawda tylko liniami konturowymi (obiekty są jakby przezroczyste), ale zachowuje się zgodnie z zasadami perspektywy i wygląda niezwykle sugestywnie zza sterów lecącego helikoptera.

Helikopter jest wolny, ale został zaprojektowany do lotów na małych wysokościach. Lot z prędkością 150 km/h na wysokości 5 metrów w urozmaiconym terenie (drzewa, góry itp.) dostarcza wielu wrażeń. W celu osiągnięcia pełniejszego złudzenia ruchu (szczególnie przy małych prędkościach) wprowadzono dodatkowe wypełnienie ekranu w postaci punktów imitujących krzaki i okruchy skał. Ułatwiają one start, ale utrudniają poszukiwanie przeciwnika, zmuszając do uważnej obserwacji wskazań instrumentów pokładowych.

Jak w każdym programie symulacyjnym i tu możemy wybrać stopień trudności gry, jednak przy pierwszym uruchomieniu program zakłada, że jesteśmy nowicjuszami i chcemy wykonać lot treningowy. Jakkolwiek w trybie tym przeciwnik nie broni się, to musisz wiedzieć, że zniszczenie wszystkich wrogich celów może zająć Ci ponad 20 godzin. Wiedz także, że prawdziwy pilot helikoptera może wydajnie walczyć w powietrzu nie więcej niż 3 godziny.

Jeśli już potrenowałeś i masz ochotę na nowe wrażenia, spróbuj polatać w chmurach, gdzie polegać można tylko na wskazaniach przyrządów, a przeciwnik strzela i trafia. Tryb lotu nocnego to już prawdziwy horror – brak horyzontu utrudnia stwierdzenie czy ziemia jest pod, czy nad Tobą, czy lecisz w niebo, czy na spotkanie z ziemią. Jeśli ciągle mało Ci wrażeń, dodaj jeszcze boczne wiatry i zawirowania. Pamiętaj, że podatność na podmuchy wiatru zależy od prędkości i wysokości. Generalna zasada brzmi: wiatr trzeba przetrzymać, a nie zwalczyć.

Poznaj teraz możliwości swojego helikoptera. Zacznijmy od uzbrojenia – składa się ono z działka kalibru 30 mm (około 100 sztuk amunicji), 38 pocisków rakietowych kalibru 70 mm i 8 rakietowych pocisków samonaprowadzających się o nazwie "piekielne ognie" i zasięgu do 3 mil. O ile przeciwnika, do którego strzelasz z działka lub rakietą, musisz utrzymywać stale na muszce, to w wypadku "piekielnych ogni" wystarczy "złapać" go tylko na chwilę w celownik – trafisz go bez wątpienia, jako że "piekielne ognie" trafiają zawsze. Z pomocą "piekielnych ogni" bardzo łatwo niszczy się przeciwnika; o wiele trudniejsze, ale i sprawiające większą satysfakcję jest zestrzelenie helikoptera wroga przy pomocy działka 30 mm lub rakiet. Pamiętaj, że czołgi zniszczyć możesz tylko rakietami lub "piekielnymi ogniami". Jeśli skończy Ci się amunicja i rakiety, możesz zawsze wrócić przez góry, co nie jest może zbyt realne, ale przecież to tylko zabawa.

Teraz kilka uwag o możliwościach ruchowych Twojego Apacza. Skomputeryzowane autostabilizatory ułatwiają pilotaż – po wszelkich ewolucjach helikopter sam wraca do normalnego poziomego lotu na wprost. Najwięcej trudności sprawia wytracenie prędkości poziomej. W prawdziwym helikopterze wymaga to około 2 mil. Dobrym sposobem na zwolnienie jest uniesienie "nosa" w górę, powoduje to jednak wzrost wysokości, a jeśli jesteś zbyt wysoko, by zobaczyć cel – musisz nurkować, co z kolei zwiększa szybkość spadania i grozi rozbiciem o ziemię. Pamiętaj, że prócz sterów (joystick) masz jeszcze "gaz".

I jeszcze parę słów o strategii walki. W danym momencie walczysz tylko z jednym helikopterem wroga, jednak zaraz po jego zniszczeniu pojawia się nowy. Może uspokoi Cię fakt, że nie jest on tak skomputeryzowany jak Twój Apacz. Nie lekceważ go jednak – pojawia się on zawsze na Twoim "ogonie" i odpowiedniej wysoko-

KILKA DOBRYCH RAD DLA POCZĄTKUJĄCEGO PILOTA

1. By szybko wystartować, daj 100% momentu obrotowego, pochyl "nos" o 30 stopni w dół, a 100 mph osiągniesz w 6 sekund.
2. Nie obawiaj się lądowania – możesz "usiąść" gdziekolwiek, a potem używając steru dokołować do lotniska.
3. W czasie ataku nie bądź zbyt wysoko, bo będziesz musiał nurkować, a to prawie zawsze kończy się roztrzaskaniem o ziemię.
4. Do czołgów strzelaj rakietami lub "piekielnymi ogniami".
5. Skutecznym sposobem na zwolnienie jest kołysanie się: to w lewo, to w prawo.
6. Żeby "ciasno" zakręcić, unieś trochę "nos".
7. Najłatwiej jest zestrzelić helikopter wroga w czasie, gdy on skręca.
8. Na początku zmień broń na "piekielne ognie", wznieś się na 512 ft. i rozejrzyj, by rozpoznać cele.
9. Terytorium i położenie wroga generuje się przypadkowo, jeśli jest zbyt trudne, przerwij program (CS i SPACE) i spróbuj jeszcze raz.

COL/TORQ – ciąg pionowy/moment obrotowy klawisz "Q" – zwiększenie; klawisz "A" – zmniejszenie

RPM/T – kolumny oznaczone 1 i 2 – obroty silników, R – wirnik, klawisz "W" – zwiększenie, klawisz "S" – zmniejszenie

FUEL – paliwo C – temperatura

CELOWNIK WIDEO – pokazuje cel znajdujący się w zasięgu broni i odległość od celu w stopach (1 stopa =0,3 m)

SPEED – prędkość pozioma w węzłach (mph)

ALT – wysokość w stopach

VSI – prędkość pionowa w stopach/s

TIME – czas do osiągnięcia celu przy aktualnej prędkości

RANGE – odległość od celu w milach (1 mila =1609 m)

ROLL AND PITCH – nachylenie płaszczyzny lotu sterowane kursorami

EKRAN NAWIGACYJNY:

liczba u góry ekranu – azymut lotu
liczba w prawym dolnym rogu – azymut "nosa"
liczba w lewym dolnym rogu – azymut celu
oznaczenie literowe w lewym dolnym rogu – symbol celu:
H – lądowisko
B – baterie p/lotnicze PRZEŁĄCZANIE – klawisz C
T – czołg
SS – helikopter

ENGS/WPNS/NAVG/TADS – wskaźniki uszkodzeń helikoptera SILNIKI/BROŃ/NAWIGACJA/CELOWNIK LASEROWY

UZBROJENIE:

- działko 30 mm
- rakiety 70 mm
- pociski samonaprowadzające "piekielne ognie"
- klawisz "P" – przełączanie celownika
- klawisz "0" – strzał

MAPA – klawisz "M"

STER KIERUNKU przy szybkości <60 mph
klawisz – CAPS SHIFT – w lewo
klawisz – Z – w prawo

ZATRZYMANIE GRY (pauza) – klawisz "H"

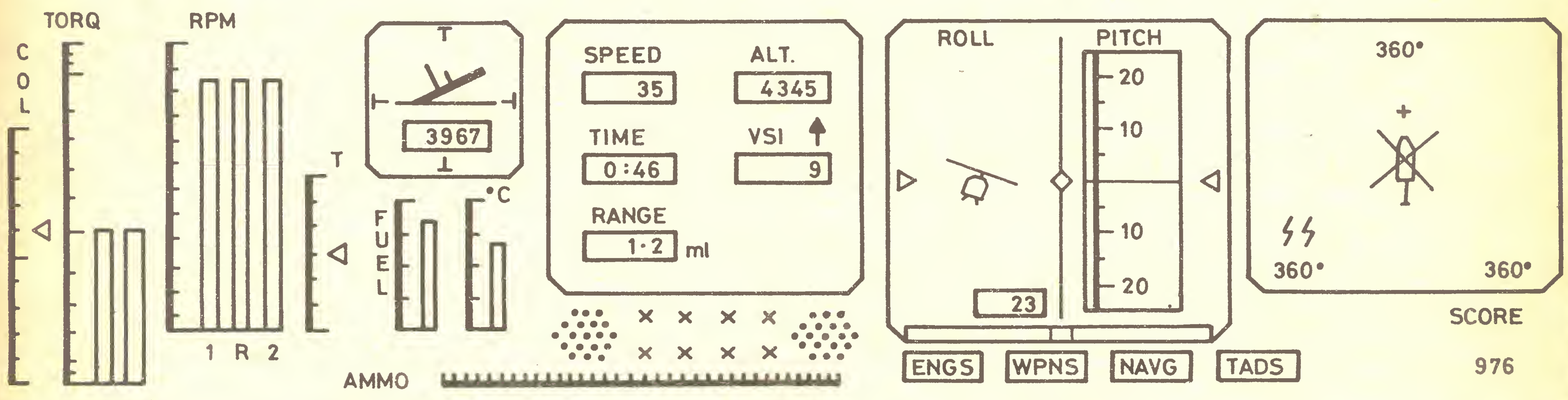
KONIEC GRY (powrót do opcji) – klawisze "CAPS SHIFT" i "SPACE"

ści, by trafić Cię swoją bronią (uzbrojony jest w rakiety o zasięgu 0,75 mili). Jeśli w trybie treningowym (gdzie przeciwnik nie strzela) nie będziesz starał się zestrzelić helikoptera wroga, to będzie on dążył do zderzenia z Tobą.

Parę słów o autorach programu. Digital Integration założyli w styczniu 1982 roku Dave Marshall i Rod Swift, zajmujący się adaptacją dla potrzeb RAF amerykańskich symulatorów lotu. DI miała początkowo zajmować się oprogramowaniem i sprzętem. We wrześniu 1983 roku Dave i Rod zdecydowali się pracować tylko na własny rachunek i porzucili pracę dla zbiurokratyzowanego ministerstwa obrony. Oprócz FIGHTER PILOT napisali NIGHT GUNNER i omawiany TOMAHAWK, a zapowiadają symulację motocykla Suzuki w wyścigu GRAND PRIX TT.

Program TOMAHAWK pisany był na dwóch mikrokomputerach CAL (systemem CP/M). CAL ma 16-bitowy procesor 6088 i Z80, co umożliwia przygotowanie kodu dla Spectrum.

ZBIGNIEW BLEWOŃSKI
wg SINCLAIR USER 3/1986



###

AMSTRAD ATAKUJE

W ubiegłym roku co kilka miesięcy pojawiały się na rynku nowe komputery firmy Amstrad, a każdy z nich był sporą sensacją cenową. Po sukcesach na rynku komputerów 8-bitowych Amstrad zamierza także rozpocząć rywalizację na rynku komputerów 16-bitowych. Pod koniec tego roku mają się pojawić pierwsze Amstrady kompatybilne z IBM PC. Będą one oferowane razem z monitorem 12" (grafika 640x220 punktów w trybie monochromatycznym), wyposażonym w antyodblaskowy ekran. W zależności od wersji Amstrad ma być wyposażony w jedną lub dwie stacje dysków elastycznych 5 1/4". Pamięć RAM ma mieć pojemność 512 K. Komputer będzie pracował w systemie operacyjnym DOS Plus (zgodnym z MS DOS) opracowanym przez firmę Digital Research. Nowy komputer ma kosztować poniżej 1000 dol.

(gs)

CAŁKOWICIE ZAUTOMATYZOWANA SIEĆ WYMIANY DANYCH

Znany brytyjski koncern Imperial Chemical Industries uruchomił system elektronicznej wymiany danych ze swymi ponad 100 klientami. Jest to sieć Tradenet firmy ICI, która pozwala na bezpośrednią łączność między komputerami wszystkich użytkowników sieci – działa ona bez względu na system operacyjny wykorzystany w komputerach.

Zamawiać więc teraz można w ICI przez 24 godziny na dobę i przez 365 dni w roku, a ICI ze swej strony może tak samo fakturować dostawy. Uzyskuje przy tym oszczędności, bowiem komputerowe wystawienie i przesłanie rachunku siecią Tradenet kosztuje tylko 4 pensy, czyli mniej niż sam znaczek, nie wspominając o kopercie, papierze, pracy maszynistki itp.

Niedogodnością takiego rozwiązania jest konieczność utrzymywania kontaktów ze wszystkimi klientami w ten sam sposób, bo inaczej może powstać bałagan. Wdrożenie systemu zajmie w ICI i u jego klientów około dwu lat. (JAL)

KONKURENT

Dominującym na rynku drukarek gigantom – firmom Epson, Star i Brother przybywa nowy groźny konkurent. Włoska firma Olivetti zapowiedziała, że jeszcze w tym roku rozpocznie wielkoseryjną produkcję drukarek, a jej oferta obejmie wszystkie typy drukarek do komputerów osobistych. (gs)

UNIX

Sześć dużych europejskich firm komputerowych: Bull, ICL, Nixdorf, Olivetti, Philips i Siemens podpisało porozumienie o utworzeniu "Open Group for UNIX Systems". Ma ono na celu koordynację i wspieranie działań zmierzających do upowszechnienia systemu zarówno wśród producentów sprzętu i oprogramowania, jak i wśród użytkowników. Prowadzone są także rozmowy z innymi firmami pracującymi nad systemem UNIX, m.in. z amerykańskim koncernem ATT i firmą software'ową Microsoft. (gs)

KONTRAKT Z KOMPUTERA

We wczesnych latach siedemdziesiątych urzędujący minister wydał zarządzenie, aby wszystkie centrale resortowe Ministerstwa Przemysłu Maszynowego dostarczały dane do centralnego ośrodka MPM (informacje o kontraktacji i realizacji w eksporcie i imporcie) na nośniku magnetycznym lub za pośrednictwem teletransmisji. Dotyczyło to również Metalexportu, centrali handlu zagranicznego, dysponującej wówczas tylko mechaniczno-elektrycznymi maszynami fakturującymi. Centrala wydzierżawiła więc terminal IBM 37-35. Bardzo szybko okazało się, że przesyłane skondensowane informacje – wystarczające dla ministerstwa – nie wystarczają dla potrzeb własnych Metalexportu. Za duży stopień agregacji. Podjęto wówczas decyzję o utworzeniu własnego ośrodka komputerowego, zakupiono minikomputer Datapoint 2200 (16 KB pamięci, dwa dyski 2,5 MB, drukarka mozaikowa, cztery monitory). Była to pierwsza, bazowa konfiguracja.

Sprzęt Datapoint już wówczas posiadał cechy nowoczesnych minikomputerów – stanowił w tamtych czasach nowość – a przede wszystkim wyposażony był w znakomite oprogramowanie. System operacyjny ukierunkowany został na przetwarzanie danych, co było dla centrali handlu zagranicznego najważniejsze. Okazał się on dla potrzeb Metalexportu bardzo przydatny.

Każdy nowy model procesora wyprodukowany przez firmę był kompatybilny z poprzednimi. Datapoint opracował system komunikacji – ARC (Dołączenie Zasobów Komputerowych). Jest to system umożliwiający łączenie poszczególnych jednostek (procesor z dyskami, z drukarką itp.) w jedną sieć, w ramach której zbiory zapisane na dyskach pracujących w sieci są dostępne dla wszystkich procesorów. Zaletą jest to, że dołączenie każdej następnej jednostki nie wymaga żadnych zmian ani w systemie operacyjnym, ani w programach aplikacyjnych. Nie trzeba więc było kupować sprzętu na zapas, wraz ze wzrostem potrzeb informacyjnych użytkownik dołącza kolejne urządzenia komputerowe.

Mówi inż. Edmund Białobłocki, szef ośrodka komputerowego:

Nasza konfiguracja sprzętu komputerowego składa się z dziewięciu procesorów pracujących w ramach sieci ARC. Uzyskaliśmy dzięki temu wystarczającą dla naszych potrzeb moc przetwarzania oraz jesteśmy zabezpieczeni na wypadek awarii. Ma to ogromne znaczenie zwłaszcza obecnie, gdy z serwisem jest nie najlepiej. Podobnie jak inne instytucje cierpimy na brak części zamiennych oraz limitów dewizowych na ich zakup... a im firma bardziej skomputeryzowana, tym większe szkody może przynieść awaria. U nas jeśli się zepsuje jakaś jednostka, pozostałe będą działać i przejmą funkcję uszkodzonej. Ani razu w Mexsie nie musieliśmy powiedzieć: informacji nie ma, bo ośrodek nie pracuje ze względu na awarię. Tak więc Datapoint sprawdził się – pracuje u nas dwanaście lat, a ja wiem, że mamy jeszcze co najmniej trzy lata na podjęcie decyzji, jaki nowy komputer kupić. Nowy – bo firma Datapoint stoi na skraju bankructwa i wycofała się z rynku krajów socjalistycznych...

Ośrodek zatrudnia 21 osób (na blisko osiemset w całym przedsiębiorstwie). Kierownik, zastępca kierownika, sekcja wprowadzania danych z kierownikiem sekcji – osiem osób. Programiści – pięć osób, wszyscy z dużym stażem w Metalexporcie, znający problematykę handlu zagranicznego. Sekcja kontroli wejścia-wyjścia – dwie osoby zajmujące się aktualizacją wszystkich słowników i kontrolą danych wyjściowych. Operatorzy systemu – cztery osoby pracujące na dwie zmiany. Czy to dużo, czy mało? Dziesięć osób stanowi minimum zatrudnienia średniej wielkości ośrodka. To ważne, bo w dobie reformy przyjęcie nagle dziesięciu osób jest praktycznie dla firm – pragnących się skomputeryzować – niemożliwe, ze względu na PFAZ przede wszystkim.

Wielu jeszcze szefów decyduje się na komputeryzację, bo modna, podejrzliwie jednak patrzy na informatyków, nie dowierza im. Dziwią się, że informatyka tyle kosztuje i niejeden myśli: po co ja się w to pakowałem?

Wprowadzać komputeryzację należy stopniowo, obejmując nią określone obszary, a dopiero później myśleć o dalszym rozwoju. Tą podstawową bazą w centrali handlu zagranicznego są zbiory posiadające informacje z wyemitowanych dokumentów statystycznych. Dokumenty te trafiają do Centrum Informatyki Handlu Zagranicznego – na ich podstawie władze orientują się w kontraktacji handlu zagranicznego. Przykładowo zsumowane dane z faktur eksportowych i importowych stanowią informację o obrotach danego przedsiębiorstwa, a w skali wszystkich central – o obrotach całego polskiego handlu zagranicznego. Stworzenie tej bazy wszechstronnej informacji o imporcie i eksporcie jest niezbędne – bez tego w ogóle nie można obecnie operatywnie zarządzać przedsiębiorstwem. Dopiero mając taką podstawę można pokusić się o inne programy, np. analizujące efektywność eksportu-importu, prowadzące sprawozdawczość w wielu przekrojach (geograficznych, w rozbiciu na kraje, na poszczególne firmy itp.).

Inż. E. Białobłocki: Założyliśmy w komputerze wszystkie cenniki eksportowanych obrabiarek z obowiązującymi na poszczególnych rynkach cenami. Dla naszej specyfiki to ważne – umożliwiło usystematyzowanie towaru eksportowego. Nazwy obrabiarek zarejestrowane są w trzech językach: po angielsku, niemiecku i polsku. Zarejestrowane są także informacje o wyposażeniu dodatkowym, częściach zamiennych, akcesoriach. Są to dane bazowe do wystawienia oferty w danym języku. Ale z punktu widzenia racjonalności przedsiębiorstwu nie opłaca się zaczynać od podsystemu CENNIKI bez bazy. Podsystem CENNIKI ma wtedy rację bytu, gdy już funkcjonuje podsystem DOKUMENTY. My dziennie emitujemy przeciętnie 380 dokumentów...

Następnym zrealizowanym w Metalexporcie jest system finansowo-księgowy. Praktycznie znaczy to, że całe finanse i księgowość są **tylko** w komputerze. Nie ma żadnych tradycyjnych kartotek. Bilanse: miesięczny i roczny, są z komputera. W efekcie dyrekcja przedsiębiorstwa w każdej chwili ma informację na temat dowolnego konta (podczas gdy w centralach nieskomputeryzowanych taka informacja dociera z reguły po miesiącu). Pozwala to na aktualną kontrolę płatności, przyspieszenie przepływu gotówki.

Jak to się stało, że tylko jeden Metalexport ma skomputeryzowany system finansowo-księgowy? Dlaczego inne centrale połamały sobie na tym zęby? Przede wszystkim w Metalexporcie zadziałała zasada przywódcy. Dyrektor naczelny i były dyrektor ekonomiczny rozpoczynali swoją karierę zawodową w działach Elektronicznego Przetwarzania Danych. Ważne było też, że w dziale EPD Metalexportu w momencie uruchomienia systemu F-K tj. w 1979 roku wszyscy mieli już duże doświadczenie. Pracownicy posiadali tzw. kulturę informatyczną. Nikt nie bał się komputera, a praca na wydrukach komputerowych była w przedsiębiorstwie rzeczą naturalną.

Do realizacji systemu F-K od razu wydzielono: jeden procesor, dwa dyski, drukarkę i cztery monitory. Wszystkie próby i testy dyrektor finansowy wraz z projektantem programistą robili na swoim komputerze – to bardzo przyspieszyło pracę. Dyrektor finansowy powiedział: "moją ambicją i wolą jest ten system wprowadzić i zrobię wszystko, by to zrealizować". Po czym przez cały rok, dzień w dzień, siedział godzinami z projektantem, również po pracy, przygotowując program...

Taki system można wprowadzić w przedsiębiorstwie tylko raz w roku, a jak się nie uda – trzeba czekać znowu cały rok. Tym terminem jest pierwszy dzień po zamknięciu bilansu rocznego. Gdy zapadła decyzja – wdrażać system – to w Metalexporcie, inaczej niż w in-

▶ 30

nych instytucjach, rozpoczęto pracę od razu na komputerze, rezygnując z równoległego prowadzenia księgowości dotychczasowym sposobem – "ręcznie". Było to zagranie va banque. W tym samym momencie, gdy został wprowadzony system komputerowy – zlikwidowano system dotychczasowy. I to w sensie dosłownym – wszystkie maszyny fakturujące zostały sprzedane. Gdy dyrektor zorientował się, że niektórzy pracownicy robili dalej obliczenia "na piechotkę", polecił zlikwidować wszystkie kartoteki.

Inż. E. Białobłocki: Dzisiaj, z perspektywy czasu, wydaje się nam, że było to jedynie słuszne rozwiązanie, wówczas jednak decyzja wyglądała na pokerową – zwłaszcza sprzedanie maszyn i likwidacja kartotek. Byliśmy skazani na sukces, odwrotu nie było. Ci, którzy próbują robić równolegle, nie są w stanie przełamać trudności, bo nagle ludziom przybywa roboty. Przygotowują dane dla komputera i liczą konwencjonalnie. Jeden ma za wiele pracy, inny za mało i nie ma tego jak rozliczyć. Robi się bałagan. A że z nadmiaru pracy można się przykładać albo do systemu tradycyjnego, albo komputerowego – mnożą się błędy i następuje dezorientacja. U nas cały pion finansowo-księgowy jest przystosowany do specyfiki systemu komputerowego. Łącznie z planem kont.

Należy mieć świadomość, że gdyby pomysł ten wyszedł z kręgów pracowniczych – nigdy nie zostałby zrealizowany; nie miałby dostatecznej siły przebicia. W Metalexporcie przeważyły determinacja i zdecydowanie dyrekcji.

Wprowadzenie systemu bazowego oznaczało odebranie monopolu wielu komórkom, których głównym celem było dotąd przygotowywanie metodą manualną sprawozdań w różnych przekrojach dla potrzeb dyrekcji. Ludzie ci – myśląc, że stanęli przed wizją utraty pracy, ignorowali komputer. Potrafili brać z ośrodka wydruki komputerowe, przepisywać w innej konfiguracji i zanosić dyrektorom. Zostało to zlikwidowane w prosty sposób – dyrektor powiedział, że nie przyjmuje innych informacji na temat kontraktacji, realizacji itp. jak tylko z komputera! Była to oficjalna promocja komputera na podstawowe źródło informacji w przedsiębiorstwie.

Po systemie F-K wprowadzono system PŁACE-KADRY-DYSCYPLINA. Od lat lista płac jest w Metalexporcie emitowana z komputera: pensje, premie, nagrody w podziale na kraj i zagranicę. Emitowane są informacje o urlopach, chorobach. Z rozszerzonego systemu KADRY korzysta szef kadr, jest to program opracowany na jego Personal na dysku stałym (żeby informacje nie były ogólnodostępne). Dzięki temu szef kadr błyskawicznie wyświetla listę pracowników np. żonatych, bezdzietnych, znających angielski i hiszpański, o stażu dłuższym niż pięć lat, którzy mają wykorzystany urlop i nie korzystali w poprzednim roku ze zwolnień lekarskich.

Jeden z następnych systemów – MEXSERWIS – rejestruje historię wszystkich obrabiarek sprzedanych zagranicą. M.in. wszystko o usterkowości i jakości maszyn. Umożliwia to informowanie producenta co się najczęściej psuje, jak się psuje. To z kolei pozwala zakładom na planowanie produkcji części zamiennych. Możliwe jest orientowanie się w kosztach serwisu, mało tego, wiadomo który monter już daną usterkę naprawiał oraz czy znajduje się blisko miejsca awarii. Takie informacje poważnie obniżają koszty serwisu.

Inż. E. Białobłocki: Zapotrzebowanie na szybką i precyzyjną informację wyświetlaną na ekranie monitora ustawionego na stanowisku pracy, leżało u podstaw decyzji zakupu komputerów osobistych zgodnych z IBM PC.

Nasze PC połączyliśmy z systemem, mały program emulacji, umożliwiający pracę jako końcówki Datapoint. Nie musieliśmy więc przeszkalać programistów w nowych językach. Pierwsze PC ustawiliśmy w 1985 roku w... gabinecie dyrektora generalnego, zgodnie z zasadą szefa. Dzięki odpowiedniemu oprogramowaniu naczelny dużo wiedział – więcej niż dyrektorzy biur. Było więc naturalne, że oni również chcieli mieć Personale u siebie. Gdy tak się już stało, zapotrzebowanie na PC zgłosili kierownicy. Nagle bowiem odwróciły się role. Do tej pory jak dyrektor chciał się czegoś dowiedzieć – wzywał kierownika. Teraz dyrektor rozmawiając z kierownikiem wszystko już wiedział i tylko pytał – dlaczego? Była to sytuacja nowa i kierownicy zrozumieli, że im też potrzebny jest własny komputer.

Wszyscy chcieli PC, ale chodziło o efektywność ich wykorzystania. Dyrektorzy zrezygnowali więc ze swoich i przestawili je do komórek ogólnoekonomicznych w swoich pionach. Ten sposób wprowadzania PC okazał się znakomity, gdyż dyrektorzy poznali możliwości sprzętu i wiedzieli, czego mogą żądać od komórek ogólnoekonomicznych.

PC oprogramowany jest w dwojaki sposób: podsystem MENAGER pozwala uzyskiwać podstawowe informacje o działalności przedsiębiorstwa (dane te są osiągalne również w dużym komputerze, który zresztą jest źródłem informacji dla PC, ale korzystanie z dużego jest nieporęczne). Drugi program to program specjalnie przygotowany dla poszczególnych działów, "na miarę" – inny np. dla działu branżowego, inny dla działu serwisu obrabiarek. Tak to w Metalexporcie pokonano lęk przed komputerem.

Następny kierunek rozwoju ośrodka informatyki w Metalexporcie to interfejs telexowy. Jest to "skrzynka telexowa", którą można połączyć PC z linią telexową. W ten sposób wszystkie przygotowane telexy mogą być nadane przez komputer po godzinach szczytu. Pracownik wywołuje wzór telexu, poprawia co trzeba i przesyła "do kolejki" do wysłania. A prawie pięćdziesiąt procent korespondencji w Metalexporcie to telexy.

Inż. E. Białobłocki: Kończymy testy i przygotowujemy się do wprowadzenia tego programu. Planujemy, że "skrzynka" będzie integralną częścią systemu MEXSERWIS. Np. jeśli wpłynie telex o potrzebie dostarczenia części zamiennych, komputer zredaguje telex do zakładu produkującego te części i nada go. Będzie też wysyłał ponaglenia, jeśli potwierdzenie nie przyjdzie we właściwym czasie. Takiego software'u nie moglibyśmy kupić na Zachodzie (zresztą oprogramowanie jest tam bardzo drogie), więc opracowaliśmy go sami. Do końca maja program ruszy. Liczymy, że przyniesie to przedsiębiorstwu zmniejszenie o około 30% kosztów telexów oraz przyspieszy i usprawni proces ich nadawania.

Dalszy rozwój komputeryzacji Metalexportu zależy tylko od finansów, bo potrzeba rozwoju jest rozumiana na wszystkich szczeblach. Nie jest to bardzo duża przeszkoda – Metalexport realizuje zakupy ze środków własnych, potrzebuje tylko zgody ministerstwa, a i z tym jakoś nie ma kłopotów. Potrzebujemy kontraktów z komputera.

TOMASZ SYPNIEWSKI

###

KOMPUTER CICHUTKI JAK MYSZKA

Każde naciśnięcie klawisza w komputerze wywołuje – poza pojawieniem się cyferki lub litery na ekranie – emisję fali o częstotliwości radiowej, którą zazwyczaj można odpowiednio czułym urządzeniem odebrać w promieniu niemal kilometra. Myśl o tym, że wszystko da się podsłuchać, jest oczywiście nieznośna dla wojskowych (dla porządku dodajmy, że także dla wielu firm cywilnych).

Dlatego Pentagon corocznie wydaje około 200 mln dolarów na program Tempest (burza), czyli "wyciszanie" komputerów stosowanych w wojsku, wywiadach i w firmach wykonujących zamówienia wojskowe.

Są dwie zasadnicze metody usuwania tego zagrożenia. Pierwsza – to stosowanie specjalnych układów scalonych, okablowania itp. Tak, aby całość emitowała słabsze sygnały radiowe niż rozwiązania standardowe. Druga metoda polega na umieszczaniu komputera w obudowie z materiału pochłaniającego w dużym stopniu fale radiowe.

Wymogi Pentagonu są utajnione, lecz i tak przywilej dostępu do nich i związane z tym zyskowne zamówienia otrzymało około 50 firm. Wiele zamówień także jest tajnych, mimo to branża domyśla się, że największe zlecenia z programu Tempest otrzymuje firma Wang Laboratories, której około 50 wyrobów spełnia te wymogi. Okoliczność, że działają one identycznie w wersji standardowej, pomogła Wang Laboratories uzyskać zamówienia Pentagonu na zwyczajne maszynki. Firma Iverson Technology Corp. zasłynęła z "wyciszenia" IBM PC dla wojska i zajmuje się tylko pracami w ramach Tempestu. Zenith uzyskał dostęp do norm drogą wykupienia małej firmy Inteq (która "wyciszała" mikrokomputery Zenitha...), co zaowocowało sprzedażą 12 000 mikrokomputerów dla wojska; Compaq sprzedaje "tempestową" wersję komputera przenośnego.

Według szefów firm mikrokomputerowych rynek niedługo eksploduje. Coraz więcej przedsiębiorstw pragnie uchronić się przed szpiegowaniem konkurencji i sprzedaż wyrobów klasy Tempest (nic nie zakazuje sprzedaży ich cywilom) szybko rośnie. Wprawdzie nadal taniej wypada zebranie danych przez postawienie paru kolejek komuś z firmy, której sekrety pragnie się poznać, ale i wywiad elektroniczny staje się coraz popularniejszy.

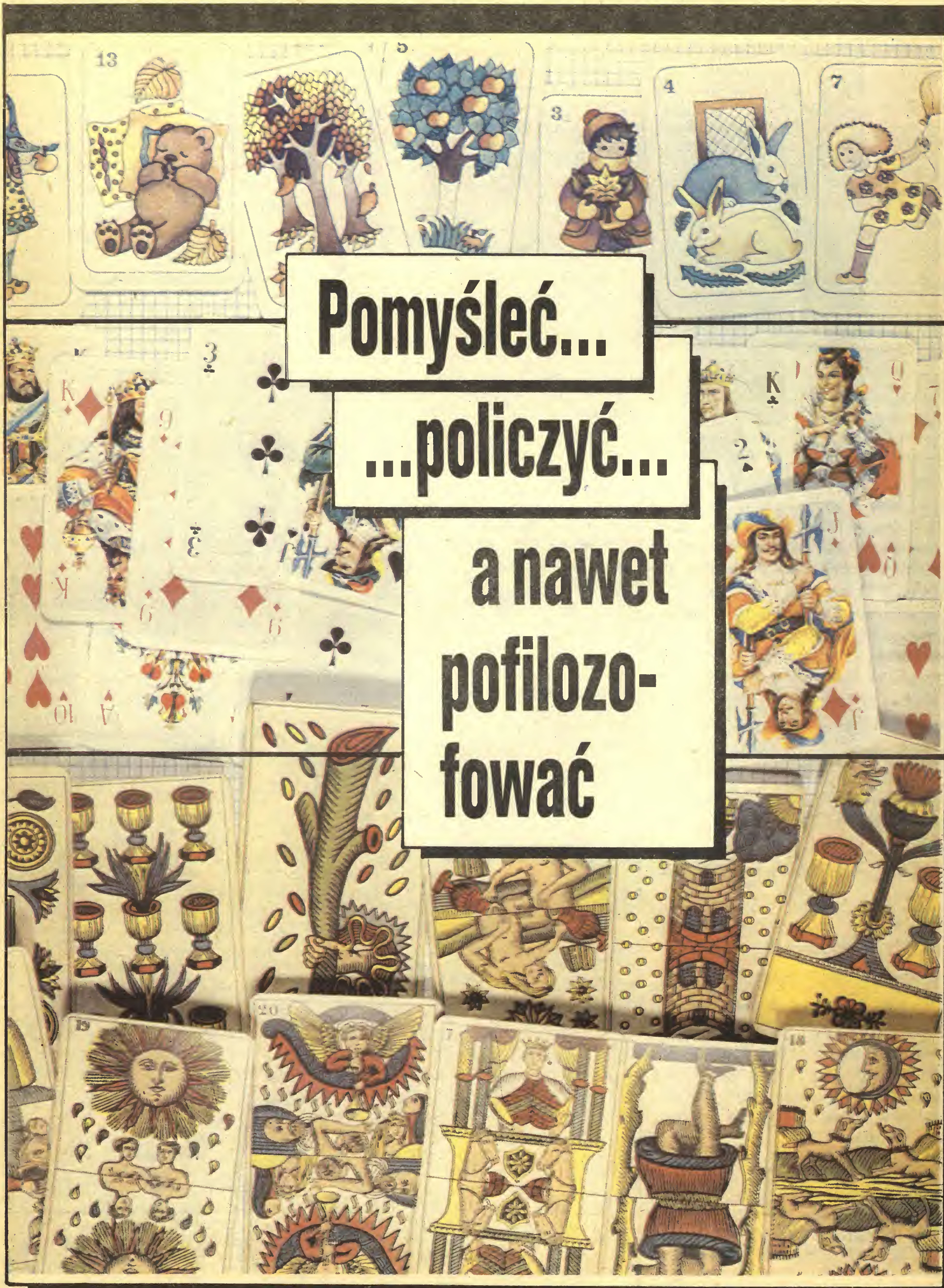
(JAL)

WARTO MIEĆ WIDEO

Nawet najlepszej jakości dyskietka nie gwarantuje, że zapisane na niej dane nie ulegną zniszczeniu na skutek mechanicznego uszkodzenia czy zakłóceń w pracy napędu. Także dane gromadzone na sztywnym dysku muszą być co pewien czas odświeżane, co wymaga wykonania kopii. Większość użytkowników komputerów sporządza więc "kopie bezpieczeństwa". Nie jest to jednak rozwiązanie zbyt wygodne z uwagi na relatywnie niewielką pojemność dyskietek i czas potrzebny na ich kopiowanie. Można oczywiście zastosować tzw. streamer – urządzenie działające podobnie jak magnetofon i umożliwiające bardzo szybki zapis na specjalnych kasetach dużej liczby danych. Streamery mają jednak dosyć istotną wadę – są bardzo drogie. Firma Alpha Microsystems, wychodząc z założenia, że większość właścicieli komputerów osobistych ma w domu także magnetowid, proponuje wykorzystanie tego urządzenia do zapisywania dużych zbiorów danych. Oferowany przez firmę dodatkowy pakiet i oprogramowanie do IBM PC umożliwia zapisanie na kasecie wideo do 80 M.

(gs)

Pomyśleć... ...policzyć... a nawet pofilozofować



Metody heurystyczne

Ostatni odcinek zakończyliśmy przykładem zastosowania algorytmu alfa-beta. Chciałbym zwrócić uwagę Czytelnika na fakt, że jest to algorytm nietrywialny i poznanie go wymaga cierpliwości. Po drugie, procedura "alfabeta()" tego programu, po niewielkich przeróbkach, może być zastosowana w innej grze. Stanowi swego rodzaju konstrukcję, do której dobudowuje się inne procedury, jak: USUŃ, WSTAW, generuj, next itd. Zalety metody alfa-beta gwarantują otrzymanie dobrych wyników, o ile reszta zrobiona będzie prawidłowo. Wobec tego możemy uznać, że kwestia przeszukiwania drzewa dla dowolnej gry została rozwiązana. Twórczego wysiłku natomiast wymagają pozostałe elementy istotne dla programu. Takim najważniejszym, i najtrudniejszym zarazem, jest funkcja oceniająca. Gdybyśmy umieli konstruować "idealne" funkcje oceniające, nie trzeba byłoby przeszukiwać drzewa gry!

W poprzednich odcinkach, przy różnych okazjach, wskazywaliśmy czynniki składające się na jakość programów grających, czyli na siłę ich gry. Wymieńmy je teraz razem (chodzi o gry, dla których nie znamy strategii wygrywającej). Są to:

- Metody przeszukiwania drzewa (minimaks, alfa-beta).
- Przeszukiwanie wstępne w celu uporządkowania węzłów przed algorytmem alfa-beta.
- Adekwatność funkcji oceniającej.

Metodom przeszukiwania poświęciliśmy już sporo miejsca. O funkcji oceniającej z kolei nie możemy powiedzieć zbyt wiele. Jest ona dla każdej gry inna i zależy przede wszystkim od wiedzy programującego o danej grze. Dlatego zespoły pracujące nad coraz lepszymi programami szachowymi mają w swych składach wybitnych szachistów.

Przeszukiwanie wstępne polega w znacznym stopniu na umiejętności rozpoznania, niemal na wyczucie, ruchów lepszych i gorszych. Stosuje się tu funkcje (procedury), które na podstawie krótkiej analizy porządkują ruchy (węzły). W tym przypadku również niezbędna jest

```
#include<stdio.h>
#define P3 3
#define P9 9
#define TRUE 1
#define FALSE 0
int Pdoc[P9],Pstart[P9],r[900];
typedef struct poz {int Pr[P9],nrsp,id,ildzieci;
                struct poz *dzieci[4], *ojciec;} POZ;
POZ *p, *px, *start;
int mem,poziom_max, l_rozw = 0,  l_ruchow = 0;
/* ilość ruchów w danej pozycji spacji */
int Tr[] ={2,3,2,3,4,3,2,3,2};
int T[4][P9]; /* możliwe ruchy */

druk(P)
int P[];
{int k,l;
for(k=0;k<P3;k++){
    for(l=0;l<P3;l++) printf(" %d ",P[k*P3+l]);
    printf("\n");}}

main()
{int spacja=-1, k,l,i,w,z,x,v,vx,rowne=1,s1,s2;

/*inicjalizacja tablicy ruchów spacji*/
for(k=0;k<P9;k++){T[1][k]=1;T[0][k]=-3;T[3][k]=0;T[2][k]=0;}
T[0][0]=T[0][1]=1;T[0][2]=T[1][0]=T[1][1]=T[1][5]=3;
T[1][2]=T[1][8]=T[2][1]=T[2][5]=T[2][7]=T[3][4]=-1;
T[2][3]=T[2][4]=3;

czyt:  printf("Podaj ciąg wejściowy (zero zamiast spacji) \n");
for(k=0;k<P9;k++){scanf("%d",&i); Pstart[k]=i;}
printf("Podaj ciąg docelowy \n");
for(k=0;k<P9;k++){scanf("%d",&i); Pdoc[k]=i;}
printf("\n Czy dobry pucel wejściowy? \n"); druk(Pstart);
printf("\n Czy dobry pucel wyjściowy? ( 1 lub 0) \n"); druk(Pdoc);
scanf("%d",&i); if(i==0) goto czyt;
for(k=0;k<P9;k++){
    if(Pstart[k]==0)spacja=k;
    if(Pdoc[k]==0) v=k;
    if(Pstart[k] != Pdoc[k]) rowne=0;}
if(rowne){printf("Sam dałeś rozwiązanie! \n"); exit();}
if(spacja ==-1){ printf("Brak spacji, czyli zera \n"); goto czyt;}

/*badanie znaku permutacji*/
Pstart[spacja]=P9; Pdoc[v]=P9;
s1=x==1||x==3||x==5||x==7;  s1+=signum(Pstart);
s2=v==1||v==3||v==5||v==7;  s2+=signum(Pdoc);
k=s1/2;  l=s2/2;
if(s1-2*k != s2-2*l) {printf("Dwa układy różnej parzystości! \n");
      printf("PRZEJSCIE NIEMOZLIWE!!!  \n"); exit();}
Pstart[spacja]=0; Pdoc[v]=0;

podaj: printf("Podaj maksymalny poziom \n");
scanf("%d",&poziom_max);
if(poziom_max<0 || poziom_max>100) goto podaj;

/*generowanie i zapis drzewa*/
p=malloc(sizeof(POZ)); start=p; p->ojciec=NULL;
for(k=0;k<P9;k++)p->Pr[k]=Pstart[k]; p->nrsp=spacja;
```

Pomyśleć...

fachowa wiedza, którą człowiek niewątpliwie posiada – przecież w trakcie gry nie zastanawiamy się nad wszystkimi możliwymi ruchami. Problem jedynie w tym, by wiedzę tę skodyfikować i zapisać w postaci algorytmu.

Powyższy wywód ma przekonać Czytelników, że w zasadzie wiemy już wszystko, mimo że wiemy tak niewiele. Reszta to już własna wiedza i inwencja. Może niektórzy czują się zawiedzeni. Jak to, czy to już cała "mądrość" komputerów? Gdzież te szokujące rozwiązania, "generalne" wyniki?

Komputer jest tylko brute-force: siła, szybkość, dokładność, i "pracowitość". Reszta to mozolna i trudna praca człowieka. Genialne programy szachowe, te z po-

```
/*główna pętla*/
generuj_nast:  l=0; v=Tr[p->nrsp];
 for(i=0;i<v;i++)
{    w=T[i][p->nrsp];
     px=malloc(sizeof(POZ));
     for(k=0;k<P9;k++) px->Pr[k]=p->Pr[k]; /*przepisywanie pucla*/
     px->Pr[p->nrsp]=p->Pr[x=p->nrsp+w];    /*przesunięcie */
     px->Pr[x]=0;                           /*   spacji    */
     if(czydobry()){ px->nrsp=x; px->ojciec=p; p->dzieci[l++]=px;}
             else free(px,sizeof(POZ));
} /*for i */
p->ildzieci=l;
E2:  if(p->ildzieci<=0) goto back;

/*Wybieranie najlepszego ruchu*/
mem=-1;  v=P9+1;
for(k=0;k<p->ildzieci;k++)
    if(v>(vx=focena(p->dzieci[k]))) {v=vx; mem=k;}
p->id=mem; /*Ten węzeł został wybrany*/

if(v==0)
 {px=p->dzieci[p->id];px->dzieci[0]=NULL;
  px->id=0;koniec(); goto cofnij;}
if(++l_ruchow<poziom_max){p=p->dzieci[mem]; goto generuj_nast;}

/*Koniec głównej pętli, początek cofania się w drzewie*/
cofnij: for(k=0;k<p->ildzieci;k++)
        free(p->dzieci[k],sizeof(POZ));
    l_ruchow--;
back: p=p->ojciec;
      if(!p){rozbrak();goto podaj;}
      w=p->id;free(p->dzieci[w],sizeof(POZ));
      p->dzieci[w]=p->dzieci[--(p->ildzieci)];
      l_ruchow--;goto E2;
}/*main end */
koniec()
{POZ *x;
int k,l,w=0;
poziom_max=l_ruchow++;
printf("Mam rozwiązanie, wciśnij klawisz  \n"); scanf("%d",&k);
for(x=start;x;x=x->dzieci[x->id])
{   printf("\n KROK  %d \n",w++);
    druk(x->Pr);}
for(l=l_rozw;l>=0;l--)
    r[l+1]=r[l];
r[0]=l_ruchow;
l_rozw++;
printf("Czy chcesz inne rozwiązania?  (1 lub 0) \n");
scanf("%d",&k);
if(k==0) rozbrak();}

rozbrak()
{int k;
if(l_rozw==0){printf("Brak rozwiązań dla %d poziomów\n",poziom_max);
          return;}
printf("Oto najszybsze rozwiązania: \n");
for(k=0;k<l_rozw;k++)  printf("%d ", r[k]);
exit();}
czydobry()
/*procedura sprawdza czy dany układ nie pojawił się wcześniej*/
{POZ *x;
int k,z;
for(x=start;x!=p;z=x->id,x=x->dzieci[z])
   {  for(k=0;k<P9;k++)
          if(x->Pr[k] != px->Pr[k]) goto en;
     return(FALSE);
en:    /*sprawdzamy dalej */
  }
return(TRUE);}

focena(a)
POZ *a;
{int v=0, k;
for(k=0;k<P9;k++)if(Pdoc[k]){
   if(Pdoc[k]!=a->Pr[k] )v++;}
return(v);}
signum(P)
int P[];
{int k,l,s=0;
for(k=0;k<P9-1;k++)
   for(l=k;l<P9;l++) if(P[k]>P[l])s++;
return(s);}
```

1	2	3
4	5	6
7	8	

8	7	6
5	4	3
2	1	

1	2	3
4	5	6
7	8	

wodzeniem rywalizujące z najlepszymi graczami, to przede wszystkim wspaniałe funkcje oceniające i porządkujące. To umiejętne skojarzenie szybkości maszyn i ludzkiej wiedzy "zaszytej" w liczbach i algorytmie.

Jednak pomysłowości człowieka nie ma końca. I tu mają trochę racji ci zawiedzeni. Jest coś więcej – może będzie coś więcej! Ale o tym, czyli o kierunkach rozwoju w informatyce, powiemy następnym razem. Tymczasem chciałbym jeszcze pozostać przy poprzednim temacie, gdyż przeszukiwanie drzew to nie tylko element służący programowaniu gier. Wszędzie tam, gdzie spośród wielu możliwości trzeba wybrać jedną, lub kilka, można zastosować tę metodę. Na przykład gdy mamy ustawić osiem hetmanów na szachownicy, by żaden nie stał w polu bicia innego, lub znaleźć najkrótszą drogę pomiędzy pewną ilością miast (tzw. problem komiwojażera). Każdy tego typu problem można rozwiązać przy pomocy drzewa: wygenerować wszystkie możliwe układy i sprawdzić, które stanowią rozwiązanie. Co jednak czynić, gdy problem jest zbyt złożony i rozwiązanie tą metodą jest niemożliwe? W takich przypadkach trzeba odwołać się do metod heurystycznych. Ogólnie, heurystyka polega na "inteligentnym" (w przeciwieństwie do brute-force) rozwiązywaniu problemów, zorientowanym na osiągnięcie celu. W tym przypadku oznacza po prostu wybieranie tych gałęzi drzewa, które na podstawie pewnej funkcji zdają się przybliżać nas do celu.

Wyjaśnimy to sobie na przykładzie tzw. puzzle-8, który będziemy nazywali puclem-8. W kwadracie 3 na 3 mamy osiem ruchomych numerków i jedno wolne miejsce, spację. Możemy przesuwać numerki pionowo i poziomo, ale tylko w miejsce spacji. Bardziej znany jest pucel-15, czyli kwadrat 4 na 4 działający na tej samej zasadzie.

Nie jest to gra w ścisłym sensie, gdyż nie ma przeciwnika, poza, być może, własną niecierpliwością. Chodzi w nim o ułożenie zadanej kolejności liczb. Numerki składają się z cyfr od 1 do 8. Jeżeli przyjmiemy, że puste miejsce to cyfra 9, wówczas przesuwanie numerków będzie permutowaniem ciągu 1, 2, 3, 4, 5, 6, 7, 8, 9. W przypadku pucla wygląda to tak:

Nam wygodniej jest utożsamiać go z ciągiem, co wychodzi na to samo. Liczba wszystkich różnych ułożeń cyfr od 1 do 9 równa jest liczbie permutacji tego ciągu, czyli 9! (9 silnia).

Kiedy w 1878 roku po raz pierwszy skonstruowano pucel, nie znano teorii związanej z permutacjami. Ogłoszono wówczas wysokie nagrody dla tych, którzy z ciągu od 1 do 9 zdołają "przejść" do zadanych z góry permutacji. Później okazało się, że były to układy "niemożliwe", do których nie można "dojść". Są dwie równe i rozłączne klasy (rodzaje) permutacji zamknięte na operacje "przejścia", co oznacza, że z żadnego układu jednej klasy nie można "przejść" do żadnego układu drugiej. Własnością, która decyduje o przynależności do klasy, jest tzw. znak lub parzystość permutacji, która zależy od tego, czy dana permutacja zawiera parzystą czy też nieparzystą liczbę inwersji (odwróceń, takich, że liczba mniejsza stoi za większą). W przypadku pucla

▶ 34

Czy tylko zgadywanka?

Gdy dwa miesiące temu zacząłem przygotowywać zgadywankę[1], nie spodziewałem się, że tak trudno jest ułożyć odpowiednie pytania, stanowiące tu gwóźdź programu. Przeszukałem atlas i encyklopedię, przewertowałem rocznik statystyczny i podręcznik geografii. Na biurku rosły sterty papieru, a ja szukałem właściwego pytania, szukałem, szukałem... Zauważyłem w pewnym momencie, że do wielu informacji wracam po kilka razy, te same pytania sprawdzam wielokrotnie. Ze zgrozą spostrzegłem też, że powtarzam tę samą procedurę szukania odpowiednich pytań dla różnych grup państw. Z obserwacji tych zrodziła się myśl, by wykonawcą tej mrówczej pracy był komputer. Wtedy jednak – artykuł musiał być gotowy na następny dzień, a ranek się zbliżał – odstawiłem komputer na bok i dokończyłem pracę "ręcznie". Teraz mogę spokojnie wrócić do tamtych myśli.

Aby zastosować algorytm wyszukiwania informacji, musiałem ułożyć drzewo binarne. Innymi słowy – w schemacie przedstawionym na rysunku w miejsce liczb podstawić odpowiednie pytania. Wyszukiwanie pomyślanego państwa odbywa się przez udzielanie odpowiedzi na kolejne pytania i przesuwanie palcem na pole w dół i w lewo, gdy odpowiedź brzmi NIE, lub w dół i w prawo – gdy damy odpowiedź TAK. Grę zaczynamy zawsze od korzenia, to znaczy z pozycji o numerze 1. Zadajemy pytanie przypisane tej pozycji, odpowiadamy np. TAK i przesuwamy się na pole o numerze 3. W ten sposób z grupy wszystkich państw wyeliminowaliśmy te, dla których odpowiedź brzmi NIE. Interesuje nas już tylko połowa państw. Powtarzamy tę procedurę dla tej połowy, potem dla odpowiedniej połowy tej połowy, potem... W końcu zostaje nam tylko jedno państwo – to, którego szukaliśmy.

Po tym wyjaśnieniu jest oczywiste, że całe "myślenie" komputera to tylko obliczanie, w którą stronę mamy się przesunąć po podaniu odpowiedzi i wypisanie na monitorze pytania odpowiadającego danej pozycji. Natomiast przypisanie pytań wierzchołkom (tak matematycy nazwaliby pola oznaczone liczbami) naszego drzewa – to już praca wymagająca pomysłowości i wiedzy: w danym wierzchołku musimy konkretną grupę państw rozdzielić na dwie w miarę równe części przy pomocy jednej tylko cechy.

Napisałem, że do wykonania tej pracy jest potrzebna wiedza i pomysłowość. Teraz już nie jestem tego pewien. Wiedza jest bezwzględnie potrzebna, to każdy widzi. Pomysłowość zaś można zastąpić cierpliwością. Zamiast wybierać od razu najlepsze do naszych celów cechy, możemy po prostu za każdym razem przeglądać całą naszą wiedzę. Pozostaje więc już tylko ostatni krok: zastąpić cierpliwość komputerem. Doszliśmy w końcu do tego, że przypisania wierzchołkom pytań, czy też cech, dokona komputer, któremu podamy naszą wiedzę.

Pierwszym dobrym pomysłem kodowania wiedzy jest użycie tablicy dwuwymiarowej. Kolumny będą "ponumerowane" obiektami, które chcemy rozróżniać, zaś wiersze tej tablicy będą odpowiadały cechom, które rozpatrujemy. Zero na przecięciu kolumny "Szwajcaria" i wiersza "dostęp do morza" oznacza, że Szwajcaria dostępu do morza nie ma. Natomiast jedynka w kolumnie "Dania" i wierszu "stolica na wyspie" oznacza, że Kopenhaga leży na Zelandii nad Sundem.

Procedura szukająca cechy najlepszej – to znaczy rozdzielającej daną grupę obiektów na dwie grupy o podobnej liczbie obiektów – przegląda tablicę wiedzy. Porównuje liczbę zer i jedynek dla każdej cechy, ograniczając się jednak tylko do zadanych obiektów. Najlepsza cecha daje połowę odpowiedzi TAK i połowę odpowiedzi NIE. Trochę gorsza, ale jeszcze dobra daje prawie w po-

33 ◆

trzeba jeszcze dodać 1 do liczby inwersji, jeżeli spacja stoi w miejscu o parzystym numerze (patrz stosowny fragment programu, który liczy znaki zadanych permutacji).

Załóżmy, że z ciągu 1, 2, 3, 4, 5, 6, 7, 8, 0 (zero zamiast spacji) chcemy "przejść" do permutacji 8, 7, 6, 5, 4, 3, 2, 1, 0. Lub inaczej, chcemy, żeby komputer znalazł najkrótsze rozwiązanie, czyli najmniejszą liczbę ruchów, jakie trzeba wykonać. Ile mamy tu możliwości do przeanalizowania? Nieskończenie wiele! Można przecież przesuwać numerki bez końca. Dlatego na początek ustalmy maksymalną liczbę ruchów, która powinną wystarczyć do znalezienia rozwiązania. Będzie to granica, do jakiej rozwijane będzie drzewo ruchów (zmienna "poziom-max" w programie). Żeby wyeliminować cykle, czyli ciągi ruchów odtwarzające poprzednią sytuację, trzeba porównywać nowy układ z każdym jego poprzednikiem (procedura "czydobry()"). Liczba możliwych posunięć waha się od 1 do 4 w zależności od położenia spacji (tablica "T[ ] [ ]" w programie).

Czy powinniśmy przeszukiwać całe drzewo, wszystkie możliwe i nie powodujące cykli układy? Taka metoda byłaby zbyt wolna i nie dostarczałaby zadowalających wyników. A jaki mamy wybór?

Wiemy, że olbrzymia część drzewa ruchów nie zbliża nas do rozwiązania i można byłoby ją pominąć. Słusznie, to jest właśnie heurystyka. Pozostaje, jak zwykle zresztą, drobiazg zapisania w postaci algorytmu metody rozpoznawania właściwych dróg (węzłów). Pomińmy na razie ten szczegół i załóżmy, że mamy już funkcję "focena()", która wybiera z nowo wygenerowanych węzłów "najlepszy". Musimy jeszcze ustalić, co zrobimy, gdy mimo zalet naszej funkcji i po wykonaniu maksymalnej liczby ruchów (zmienna "poziom-max") nie otrzymamy pożądanego rozwiązania. Chodzi o fragment w programie rozpoczynający się etykietą "cofnij". Jest to ważna technika programowania, tzw. backtracking, czyli wycofywanie. Wycofujemy ostatnio wykonane ruchy, które nie doprowadziły nas do celu i zamiast nich wykonujemy inne – jak w metodzie prób i błędów.

Zajmijmy się teraz poważniej funkcją odpowiedzialną za wybór węzła. Stanowi ona podstawę metody heurystycznej i od niej zależy skuteczność znajdywania rozwiązań. Tak jak w przypadku innych omawianych tu funkcji, nie jest łatwo ją wymyślić. Funkcja "focena()" jest zbyt elementarna. Działa na prostej zasadzie zliczania numerków stojących na pozycjach docelowych. Oznacza to, że spośród wszystkich możliwych ruchów w danej pozycji wybiera taki, który ma największą liczbę "dobrze stojących" numerków. Niezbyt często prowadzi ona do celu bezpośrednio, bez wycofywania. Do poziomu 18, czyli do osiemnastoruchowych różnic pomiędzy permutacjami, znajduje w miarę szybko rozwiązania. Jednak – na czym mi najbardziej zależało – nie jest w stanie dać rozwiązania następujących permutacji:

Skądinąd wiem, że jest ono 30-ruchowe. Próbowałem kilku modyfikacji tej funkcji – na przykład uzależniałem generowanie nowych węzłów od numeru poziomu i pewnej zadanej stałej. Udało mi się osiągnąć rozwiązanie nieco innego układu, również 30-ruchowe i to po 40 sekundach (na IBM PC), ale to wszystko. Dlatego apeluję do Czytelników, którzy będą mogli poświęcić temu więcej czasu, by próbowali wymyślić taką funkcję heurystyczną, która w rozsądnym czasie prowadziłaby do rozwiązania podanych wyżej permutacji.

JANUSZ KRASZEK

```
 10 REM  SYSTEM EKSPERTOWY ?
 20 REM Dana jest tablica
 25 REM  Wiedzdy  W
 30 REM oraz zmienne  ilcech
 35 REM (ilosc cech) i ilob
 40 REM (ilosc obiektow)
 43 REM
 45 REM liczymy szerokosc drzewa
 50 LET szer=1
 60 IF ilob<=szer THEN  GO TO 90
 70 LET szer=2*szer
 80 GO TO 60
 90 DIM T(szer-1): REM tablica pytan
100 DIM z(ilob): REM pomocnicza tablica obiektow
110 REM przypisujemy pytania do wierzcholkow
120 FOR n=1 TO szer-1
130 GO SUB 200: REM ustalamy  obiekty
140 GO SUB 500: REM wybieramy  pytanie
150 NEXT n
160 STOP
200 REM Podpogram ustalajacy
205 REM badana grupe obiektow.
210 REM wynik w tablicy z; 1 - obiekt trzeba badac
215 REM
220 REM na poczatku wszystkie
230 FOR i=1 TO ilob
240 LET z(i)=1
250 NEXT i
260 LET rob=n
270 IF rob=1 THEN  RETURN
280 REM ustalamy odpowiedz
290 LET odp=rob-2*INT (rob/2)
300 LET rob=INT (rob/2)
310 REM odrzucamy zbedne obiekty
320 FOR i=1 TO ilob
330 IF W(T(rob),i)<>odp THEN  LET z(i)=0
340 NEXT i
350 GO TO 270
500 REM Podprogram wybierajacy najlepsze pytanie.
510 REM wynik programu w T(n)
515 REM
520 REM liczymy ilosc obiektow
530 LET p=0
540 FOR i=1 TO ilob
550 LET p=p+r(i)
560 NEXT i
570 IF p>1 THEN  GO TO 600
580 LET t(n)=0: REM flaga konca galezi
590 RETURN
600 REM przeszukiwanie Wiedzy
610 LET min=p*p+1
620 FOR i=1 TO ilcech
630 LET jeden=0
640 FOR j=1 TO ilob
650 IF z(j)=0 THEN  GO TO 670
660 LET jeden=jeden+W(i,j)
670 NEXT j
680 REM wybor lepszego pytania
690 LET zero=p-jeden
700 LET lepszy=zero*zero+jeden*jeden
710 IF min<lepszy THEN  GO TO 740
720 LET min=lepszy
730 LET T(n)=i
740 NEXT i
750 RETURN
```

łowie odpowiedź TAK. Jak jednak wybrać z dwóch trochę gorszych cech lepszą? Sposób na to daje matematyka elementarna. Otóż jeżeli mamy zadaną liczbę obiektów *p*, to badamy wartość bezwzględną różnicy liczb *a* i *p-a*. Im ta wartość jest mniejsza, tym podział wyznaczony przez liczbę *a* jest lepszy.

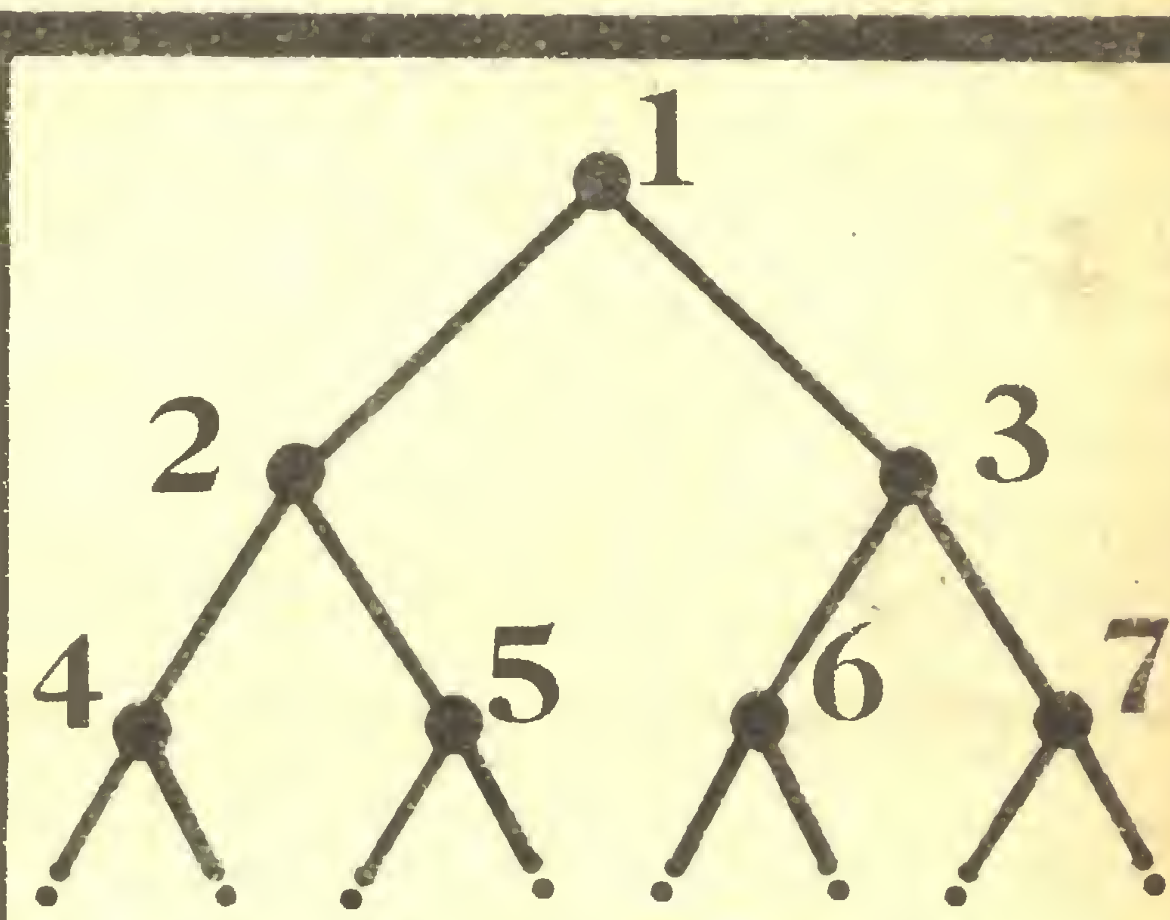
Mając opisaną wyżej procedurę możemy przystąpić do konstruowania programu dopisującego do wierzchołków naszego drzewa odpowiednie pytania. Musimy tylko napisać jeszcze jedną procedurę generującą numery obiektów, które należy rozpatrywać będąc w danym wierzchołku. Nieocenioną informację daje tu numer tego wierzchołka. Otóż zapisując ten numer w układzie dwójkowym dostajemy informację, jak należy odpowiadać na pytania przypisane wcześniejszym wierzchołkom, by dotrzeć do naszego miejsca. Oczywiście zero to odpowiedź NIE (czyli gałąź lewa), a jedynka to odpowiedź TAK.

Jeżeli po moich wyjaśnieniach wydaje się, że taki program jest bardzo trudny i skomplikowany, to proszę przeanalizować zamieszczony tu listing programu w Basicu. Zakładam, że wcześniej została zadeklarowana tablica "W/ilcech, ilob/" reprezentująca naszą wiedzę, przy czym zmienne "ilcech" i "ilob" zawierają odpowiednio liczbę cech i obiektów, które rozpatrujemy. Program tworzy tablicę T tak, że T (i) zawiera numer pytania, które musimy zadać w wierzchołku o numerze i. Proszę o wyrozumiałość – aby nie zaciemniać algorytmu, pominąłem wszelkie zabezpieczenia.

Zapewne podczas analizy tego przykładu Czytelnik ze zdziwieniem zauważy, że do wyboru najlepszej cechy użyłem nie wartości bezwzględnej różnicy liczb zer i jedynek, lecz sumy ich kwadratów. Efekt tego działania jest taki sam: im suma kwadratów liczb *a* i *p-a* mniejsza, tym podział lepszy, a sposób ten jest bardziej uniwersalny.

Niechaj przykładowo obiektami będą nie państwa, lecz grzyby. Wygodniej dla człowieka jest odpowiadać na jedno pytanie: „jaki jest kolor znalezionego właśnie grzyba?", niż na 18 lub 50 pytań w postaci "czy ten grzyb jest jasnoliliowy?". Oczywiście dla takiego rozpoznawania grzybów nasze drzewo będzie trochę inne. Z każdego wierzchołka będzie wychodziło być może więcej gałęzi – tyle, ile jest kolorów grzybów, kształtów kapelusza itp. Zasada jednak pozostanie ta sama. Będziemy wybierali taką cechę, która da najrówniejszy podział na pewną, nieznaną z góry, liczbę grup. Tutaj najmniejsza suma kwadratów liczebności poszczególnych grup też wskaże najlepszy podział (proszę przemyśleć dowód lub sprawdzić na przykładach, że tak jest rzeczywiście!).

Drzewo rozpoznawania grzyba jest bardzo skomplikowane. Bardziej skomplikowany będzie również program dopisujący pytania do wierzchołków. O skali trudności może świadczyć fakt, że dopiero po ustaleniu pytania właściwego dla danego wierzchołka możemy określić liczbę odgałęzień. W tym miejscu należy odpowiedzieć sobie na bardzo konkretne pytanie: czy jest



sens zajmować się takimi problemami? Oczywiście jest. Nie chodzi bowiem o zgadywanki, pokazy na prywatkach czy zabezpieczenia przed muchomorem sromotnikowym. Sprawa takiego programu nabiera znaczenia dla lekarzy do automatycznego stawiania diagnozy, dla prawników, chemików, historyków – wszędzie tam, gdzie mamy już olbrzymi zasób wiedzy i chcemy na jej podstawie rozpoznawać obiekty i zjawiska.

Program zaprezentowany w tym artykule jest "cieniem wielkiej góry". Ta góra to tzw. systemy ekspertalne: programy komputerowe, analizujące duże bazy danych reprezentujące naszą wiedzę. Na podstawie tej analizy system ekspertalny ustala pewne reguły postępowania i sposoby korzystania z nagromadzonej wiedzy. Systemy takie są intensywnie badane w wielu krajach. W wielu miejscach już istnieją i doskonale się sprawują.

Na zakończenie więc pytanie. Czy umiejętności korzystania ze zdobytej wiedzy nie zwykliśmy nazywać inteligencją?

LESZEK RUDAK

[1] patrz Komputer 3'86

Visicalc

Dla tych, którzy pragnęliby wykorzystać komputer do czegoś poważniejszego niż gry, proponuję zapoznanie się z programem kalkulacyjnym "VisiCalc".

Nie bez powodu wybrałem "VisiCalc" – do tej pory sprzedano ponad 0,6 mln kopii tego programu, co czyni go niemalże standardem przemysłowym dla komputerów osobistych. Wszystkie późniejsze programy kalkulacyjne to mniej lub bardziej rozbudowane wersje "VisiCalc". Należą do nich: "Multiplan", "SynCalc", "Mars" czy polski "Tab-Plan".

Programy kalkulacyjne są programami ogólnego przeznaczenia, używanymi przede wszystkim w planowaniu finansowym. Z reguły tego typu działanie wymaga częstego przeliczania wyników. Tradycyjna metoda – polegająca na ręcznym wpisywaniu danych do odpowiedniego formularza i przeliczaniu ich przy pomocy kalkulatora – jest zbyt wolna i nieefektywna, stąd duże prawdopodobieństwo nieoptymalnych rozwiązań. Stało się to powodem opracowania w 1979 r. przez D. Bricklin'a i R. Frankston'a programu "VisiCalc" (ang. instantly calculating electronic worksheet – elektroniczny arkusz kalkulacyjny).

Arkusz zorganizowany jest jako macierz 63 kolumn (ponumerowanych od A do BK) i 254 wierszy. Każda pozycja (komórka) w tej macierzy przedstawionej na ekranie odpowiada polu w pamięci komputera, które może zawierać daną numeryczną, tekstową lub formułę obliczeń. I tak, jeśli pole A1 (A – numer kolumny, 1 – numer wiersza) zawiera daną tekstową "ZYSK", tekst ten pojawi się na ekranie w komórce A1. Podobnie jest z formułami zawartymi w pamięci komputera, np.: jeśli pole B6 zawiera formułę +B2+B4, rezultat dodania zawartości pól B2 i B4 pojawi się na ekranie w komórce B6 (oczywiście formuły mogą być bardziej rozbudowane).

Elastyczność programu polega na tym, że komputer pamięta wszystkie dane, formuły i obliczenia, których dokonuje użytkownik. Tak więc, jeśli zmieni on jakąkolwiek daną lub formułę, wszystkie powiązane dane zostaną natychmiast ponownie przekalkulowane. Pozwala to nie tylko szybko poprawić ewentualne pomyłki, ale i badać różne warianty (ang. what if – co się stanie jeśli).

"VisiCalc" ma wiele możliwości ułatwiających posługiwanie się nim. Do nich należą m.in.:

- replikowanie tych samych danych i formuł w całym lub części wiersza (kolumny),
- wstawienie pustego wiersza lub kolumny,
- "zmazanie" całego lub części wiersza (kolumny),
- wbudowane funkcje standardowe (wartość średnia, suma, minimum, maksimum, sinus itp.),
- zapamiętanie części lub całego arkusza na dyskietce,
- wpisanie zawartości części lub całego arkusza na drukarce,
- przeglądanie arkusza we wszystkich kierunkach (przy pomocy klawiszy do sterowania kursorem),
- podział ekranu na dwa "okienka" (ang. window).

Posługiwanie się programem jest stosunkowo łatwe – proste obliczenia można wykonywać niemal bez żadnego przygotowania. Rozwiązywanie skomplikowanych zagadnień wymaga nie tylko opanowania skrótów wszystkich komend, ale i sporego doświadczenia.

Zastosowanie programu "VisiCalc" prześledzimy na bardzo prostym, typowym przykładzie planowania wyników finansowych w jakimś okresie. Niech będzie nim I kwartał roku (styczeń, luty, marzec). Zadaniem planisty jest tak ustalić wielkość produkcji i koszty, aby osiągnąć maksymalny zysk.

KROK 1

Czynności wstępne polegają na załadowaniu programu do pamięci komputera oraz sformatowaniu dyskietki niezbędnej do dalszej pracy. Ponieważ oryginalna wersja "VisiCalc" przeznaczona jest dla starego modelu Atari 800, należy najpierw wczytać program symulujący działanie modelu 800 na Atari 800XL (np. "Translator" lub "Osload").

Dyskietkę możemy sformatować wcześniej przy użyciu DOS 2.0 lub 2.5 albo też posłużyć się komendą "VisiCalc", przeznaczoną do formatowania (/SI). Uwaga! Pamiętać należy, że formatowanie kasuje dotychczasową zawartość dyskietki.

KROK 2

Jest nim wprowadzenie wszystkich danych tekstowych (ang. label). Po przesunięciu kursora do wybranej komórki (klawisze oznaczone strzałkami), napisaniu tekstu i wciśnięciu RETURN, tekst zostaje zapamiętany w danej komórce. Najpierw jednak warto zwiększyć rozmiar komórki do 10 miejsc (standardowy wynosi 9). Podajmy więc: /GC 10 RETURN – i gotowe. Jeśli wprowadzany tekst nie mieści się w jednej komórce, możemy jego część przerzucić do komórki sąsiedniej, jak to uczyniono w kolumnach A i B.

Do "narysowania" linii przerywanej w wierszu 2 najwygodniej zastosować replikowanie danych i to podwójne. Ustawmy kursor w komórce A i wciśnijmy /– –, co spowoduje wypełnienie komórki wybranym znakiem tj. "–" (kopiowanie danej w komórce). Nie zmieniając położenia kursora wciskamy klawisze /R, w linii komend ukaże się komunikat:
REPLICATE: SOURCE RANGE OR RETURN
Ponieważ chcemy skopiować zawartość jednej komórki, a nie ich ciągu (range), wciskamy RETURN, co spowoduje wyświetlenie komunikatu:
REPLICATE: TARGET RANGE
Dzieje się tak, ponieważ potrzebna jest informacja, w jakim ciągu komórek zawartość A2 ma być powtórzona. W naszym przypadku są to oczywiście komórki między B2 i G2. Współrzędne tego ciągu podajemy naciskając B2.G2 i RETURN; w linii wejścia "." (kropka) zastąpiona zostanie trzema kropkami, tak więc podany ciąg wygląda następująco: B2...G2. Po wykonaniu tych czynności w wybranej części wiersza 2 mamy "narysowaną" linię przerywaną.

Podczas pracy z arkuszem wygodnie jest mieć nagłówki umiejscowione na ekranie; podczas przewijania (ang. scrolling) są one stale widoczne, co pozwala na szybką orientację w gąszczu liczb. Służy do tego komenda /T, po wykonaniu której pojawi się komunikat:
TITLES: HVBN
Oznacza on, że wiersze powyżej kursora (H), kolumny na lewo od kursora (V), wiersze powyżej i na lewo od kursora (B) będą umiejscowione na trwałe. "N" – kasuje działania komendy. W naszym przypadku wystarczy kursor umieścić w komórce B1, po czym wcisnąć /TB.

KROK 3

Wprowadzenie wszystkich danych i formuł (ang.value). Wprowadzanie danych jest bardzo proste: należy ustawić kursor, wpisać daną (w naszym przypadku maksymalnie 10 cyfr) i nacisnąć RETURN.

38

	Styczen	Luty	Marzec	Suma	$rednio
Koszt jednostkowy	250	250	400		300
Ilo$c produkcji	50000	50000	90000	190000	63333.333
Koszty ogolem	12500000	12500000	36000000	61000000	20333333.
Cena	300	300	480		360
Warto$c spzedazy	15000000	15000000	43200000	73200000	24400000
Zysk brutto	2500000	2500000	7200000	12200000	4066666.7
Obciazenie zysku	1875000	1875000	2016000	5766000	1922000
Zysk netto	625000	625000	5184000	6434000	2144666.7

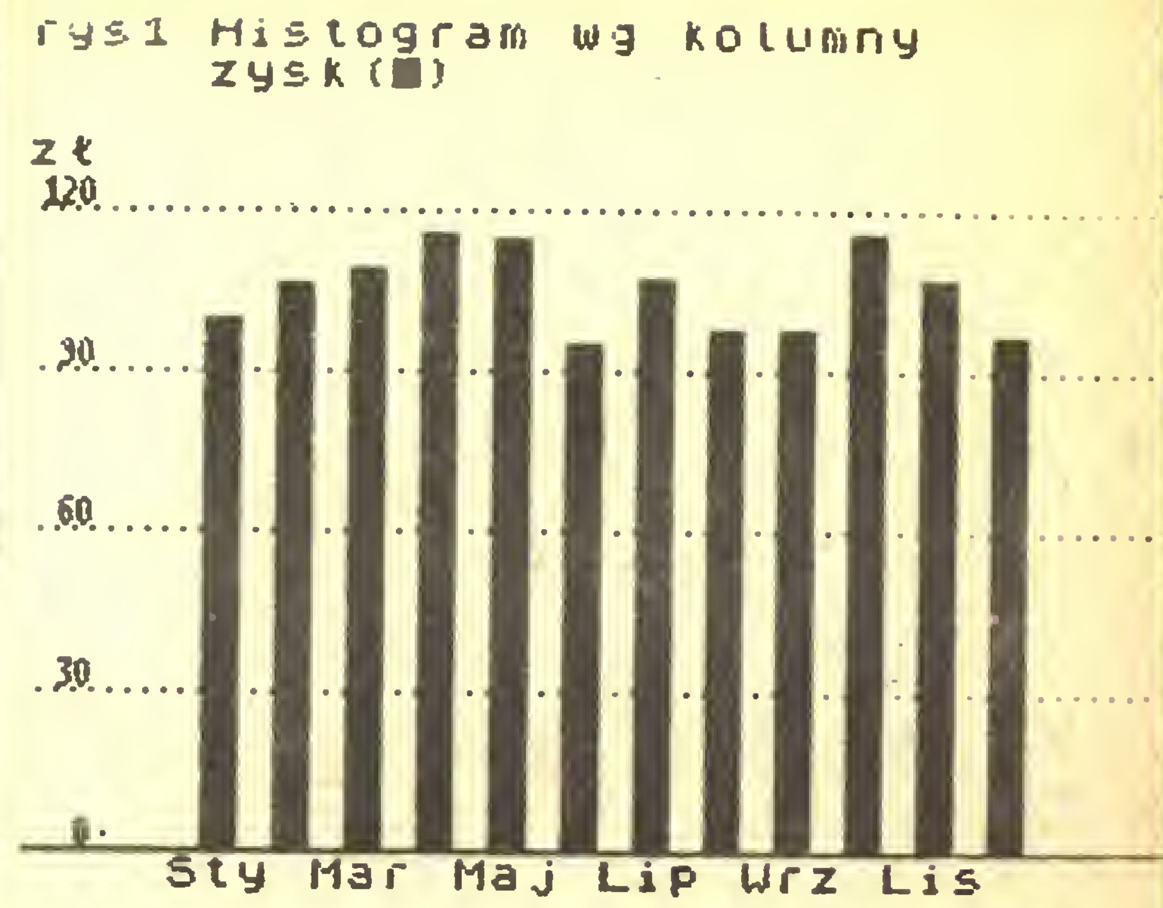
Omnicalc 2

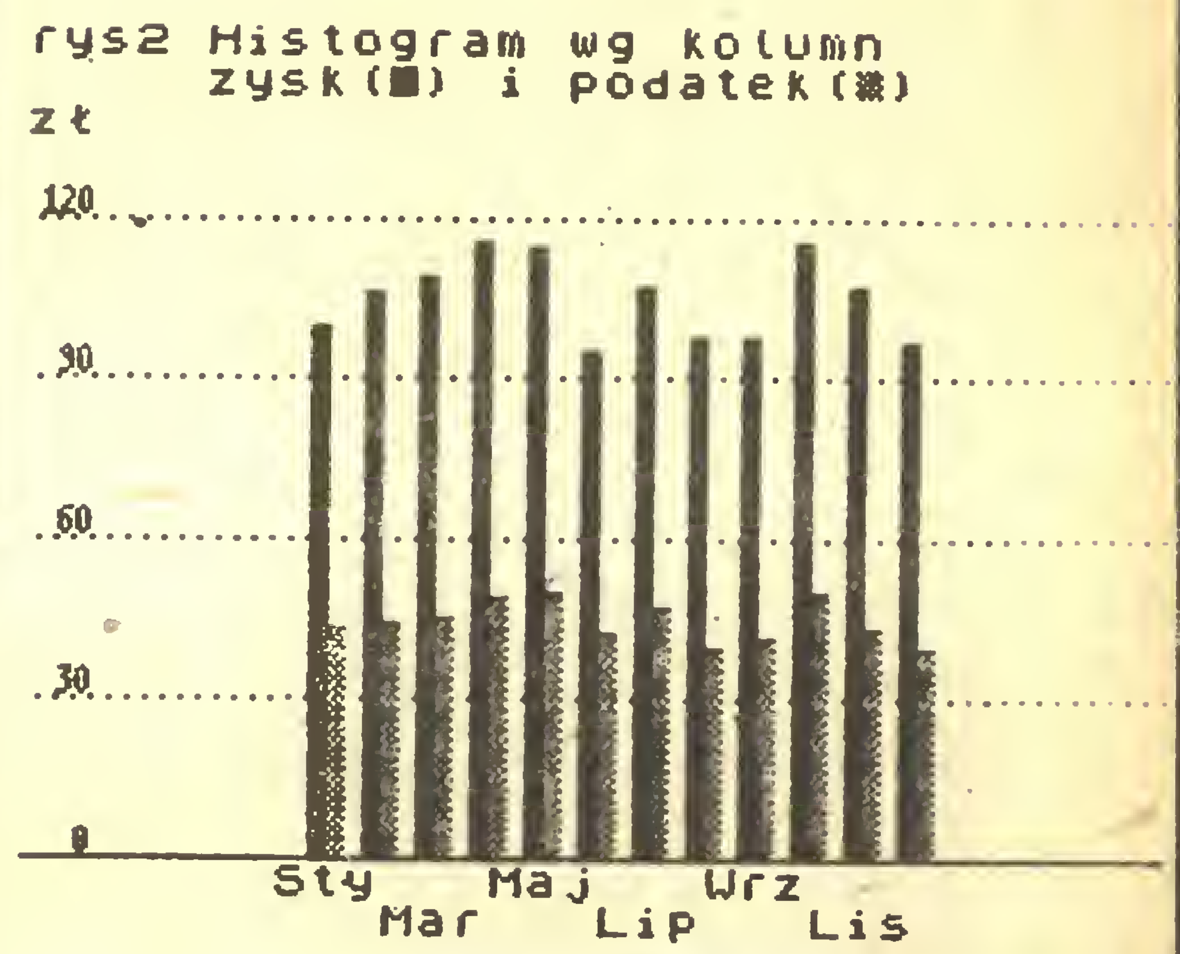
Każdy musiał choćby raz sporządzić jakąś tabelkę: wpisać w wiersze lub kolumny dane, na podstawie których obliczane były następnie dalsze elementy tabeli. Kolumny i wiersze mogły być sumowane, dzielone, mogły być także, co jest żmudniejsze, liczone udziały procentowe. Niekiedy trzeba było przedstawić wyniki w formie graficznej. Wszystko to wymagało dużego nakładu pracy, a jeżeli po pewnym czasie okazywało się, że w danych był błąd lub że należało inaczej przeliczyć pewne elementy, to sprawa zaczynała być nużąca.

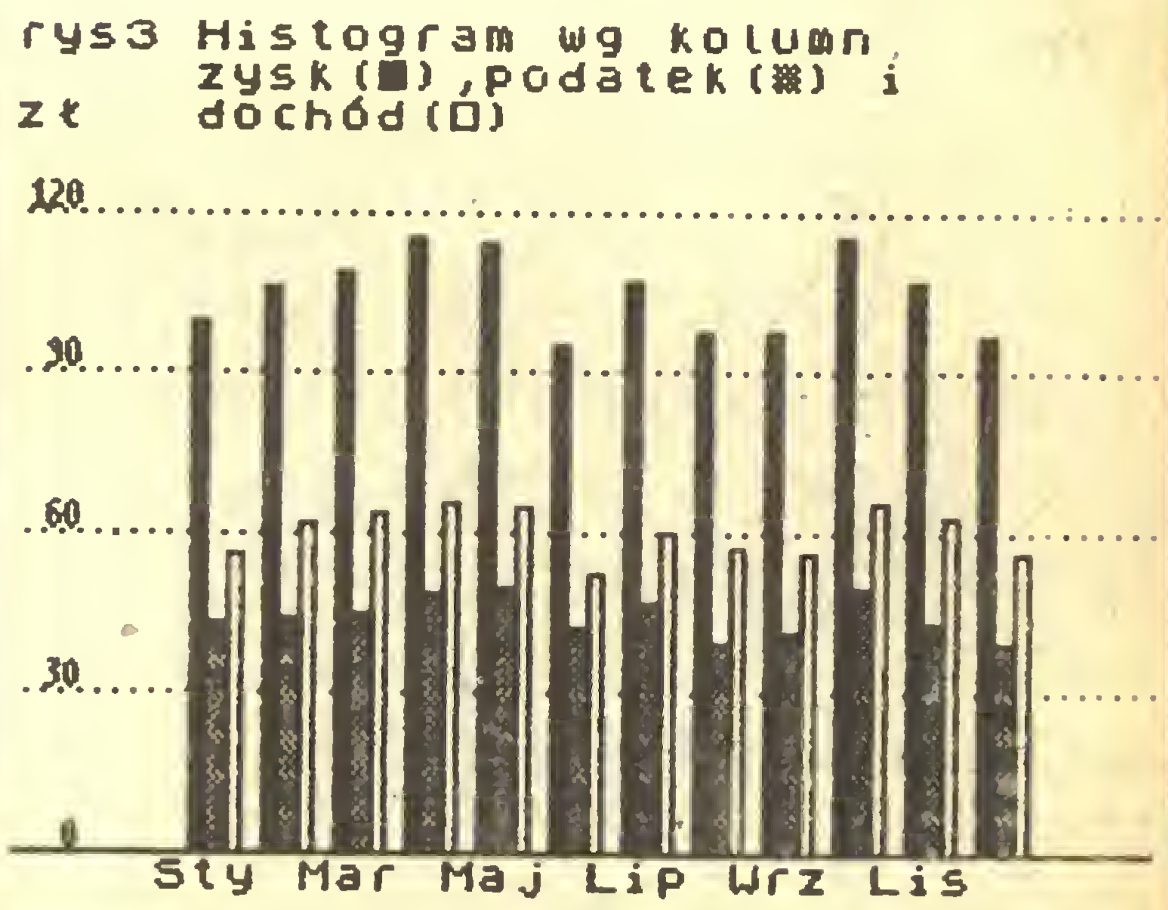
We wszystkich tych problemach pomóc może mikrokomputer wraz z odpowiednim programem. Ten program to tzw. spreadsheet, czyli wszechstronny program kalkulacyjny, nazywany po polsku niekiedy arkuszem elektronicznym bądź po prostu tabelą (w "Komputerze" J. Tatarkiewicz lansuje termin "płachta" – dop. red.). Prekursorem tego typu programów mikrokomputerowych był "VISICALC". Owiany niemalże legendą, stał się już standardem w oprogramowaniu użytkowym komputerów osobistych, tych profesjonalnych jak i tych mniejszych, domowych. "VISICALC" po raz pierwszy wyprodukowany został przez firmę VISICORP dla pierwszego mikrokomputera osobistego – Apple. W miarę rozwoju rynku mini– i mikrokomputerowego pojawiły się coraz to nowe implementacje i bardziej rozbudowane wersje programu, np. "MULTIPLAN", "SUPERCALC" czy "LOTUS 1–2–3".

Także najpopularniejszy w naszym kraju komputer domowy Sinclair ZX Spectrum ma w swojej bibliotece oprogramowania odmianę programu "VISICALC": "OMNICALC 2" opublikowane w 1984 r. przez firmę MICROSPHERE.

Program umożliwia zbudowanie tabeli o wymiarach do 99 kolumn (numerowanych od 1 do 99) i 250 wierszy (oznaczonych symbolami literowymi od a do ip). W sumie jednak liczba elementów (iloczyn liczby wierszy i kolumn) wynosić może około 4200. Oznaczenia wierszy i kolumn umożliwiają jednoznaczne określenie położenia każdego elementu tabeli np. a9, ab35. Wiersze i kolumny, oprócz oznaczeń nadawanych automatycznie przez program, mogą mieć nagłówki. Są one wyświetlane z odpowiednią kolumną czy wierszem. Dzięki temu zawsze wiadomo, w którym miejscu tabeli się znajdujemy, co przy większych rozmiarach arkusza nie jest bez znaczenia. Na ekranie komputera mieści się bowiem tylko jego fragment obejmujący 3 kolumny i 15 wierszy oraz 1 kolumnę i 2wiersze zawierające wspomniane nagłówki. Poszczególne elementy tabeli mogą zawierać: dane numeryczne, informacje tekstowe, zależności algebraiczne mówiące o sposobie obliczenia zawartości, mogą też być puste. Przy definiowaniu zależności algebraicznych można używać wszystkich matematycznych funkcji Spectrum. Możliwe jest także używanie wyrażeń warunkowych. Wartości z tabeli mogą być w bardzo prosty sposób przedstawione w formie graficznej (wykresy słupkowe). Do tego celu służy specjalny bufor (nazywany tutaj obszarem roboczym – workspace), do którego przenosi się wartości przeznaczone do narysowania. Drugą funkcją obszaru roboczego jest komunikacja z innymi tabelami, gdyż istnieje możliwość nagrania go oddzielnie na magnetofonie lub microdrive.







omni calc NR.KOL.	SPRZE--DAŻ (1)	KOSZT WŁASNY (2)	ZYSK (1)-(2) (3)	PODATEK 45 % (4)	DOCHÓD (3)-(4) (5)
JAN	1051	951	99	43	56
FEB	1118	1012	106	44	62
MAR	1146	1037	108	45	63
APR	1216	1100	115	49	66
MAY	1210	1095	114	49	65
JUN	1001	906	95	42	52
JUL	1127	1019	107	46	60
AUG	1027	929	97	40	57
SEP	1029	931	97	41	56
OCT	1218	1102	115	49	66
NOV	1130	1022	107	43	63
DEC	1018	922	96	40	56
SUMA	13291	12026	1256	531	722

"OMNICALC 2" jest bardzo prosty w użyciu. Każde działanie, jakie chce podjąć użytkownik, jest poprzedzane zapytaniem ze strony komputera. Obsługa klawiatury podczas wprowadzania komend jak i danych jest tak zaprogramowana, że w przypadku wykrycia błędu polegającego na naciśnięciu nieodpowiedniego klawisza, jest on sygnalizowany odmiennym dźwiękiem. Pozostałe błędy sygnalizowane są odpowiednimi komunikatami.

W większości przeróżnych tabelek pewne grupy wartości obliczane są w ten sam sposób, np. wartości z kolumny podatku zawsze stanowią pewien procent wartości z kolumny dochodu. Nasz program z łatwością

➧ 39

36 ◀

Niewiele trudniejsze jest wprowadzenie formuł. Formuły określają sposób obliczania wartości danej komórki z wykorzystaniem informacji zawartych w innych komórkach. Formułę należy rozpoczynać cyfrą, kropką dziesiętną ".", znakiem dodawania "+", odejmowania "–", lewym nawiasem „(", znakiem równości "=" lub znakiem "   ". W przeciwnym razie komputer będzie oczekiwał na wprowadzenie tekstu.

A oto jak przedstawiają się nasze formuły:

- koszty ogółem = wielkość produkcji × koszt jednostkowy, więc w komórki C5, D5, E5 należy wpisać odpowiednio: +C3×C4, +D3×D4, +E3×E4;
- cena = 1.2×koszt jednostkowy, więc w komórki C6, D6, E6 wpiszemy odpowiednio 1.2×C3, 1.2×D3,1.2×E3;
- wartość sprzedaży = wielkość produkcji × cena;
- zysk brutto = wartość sprzedaży – koszty ogółem, więc w komórki C8, D8, E8 wprowadzimy odpowiednio +C7–C5, +D7–D5, +E7–E5;
- na obciążenie zysku wymyśliłem następującą formułę:

obciążenie zysku = zysk brutto – zysk brutto × zysk brutto /10000000, stąd w komórki C9, D9, E9 należy wprowadzić odpowiednio:

+C8–(C8×C8/10000000), +D8–(D8×D8/10000000), +E8 – (E8×E8/10000000);

- zysk netto = zysk brutto – obciążenie.

Pozostaje jeszcze wprowadzenie formuł w kolumnach F i G, w których określamy sumę odpowiednich komórek z kolumn C, D, E oraz ich wartość średnią. Wprowadzenia formuły na sumę można dokonać dwoma sposobami: albo przez proste dodatnie zawartości komórek, albo z wykorzystaniem wbudowanej funkcji sumowania zawartości ciągu komórek. Tak więc wprowadzenie np. w komórkę F4 formuły + C4+D4+E4, jest równoważne formule SUM(C4...E4). Powyższych formuł nie wprowadzamy do komórek F3 i F6, jako że suma kosztu jednostkowego i ceny w kolejnych miesiącach nie ma ekonomicznego sensu.

W celu obliczenia wartości średnich skorzystamy ze standardowej funkcji AVERAGE. Tak więc formuła na obliczenie np. zysku średniego (kursor ustawiony w komórce G10) wygląda następująco:

AVERAGE (C10...E10)

Po tym kroku (jeśli wprowadzimy wartości formuły i podane przeze mnie) arkusz wygląda jak poniżej:

KROK 4

W kroku tym dokonywana jest właściwa kalkulacja w cyklu: wprowadzenie danych – przeglądanie wyników – ponowne wprowadzenie danych, aż do uzyskania satysfakcjonującego rezultatu. Wiemy już, że rezultatem tym powinien być maksymalny zysk netto uzyskany w I kwartale. By go obliczyć, "planista" może zmieniać koszty jednostkowe i wielkość produkcji. Przyjmijmy, że obie te zmienne mogą się zmieniać odpowiednio w przedziałach: 200÷500 zł za szt.

Zadanie ma jedno optymalne rozwiązanie (matematycznie łatwe zresztą do wyliczenia), natomiast kombinacji kosztów i wielkości produkcji dających rozwiązanie jest całkiem sporo.

KROK 5

Zapisanie arkusza z uzyskanym rozwiązaniem na dyskietce w celu ewentualnego dalszego wykorzystania. Po wciśnięciu /SS i podaniu nazwy arkusza, wszystkie informacje zostaną umieszczone na dyskietce.

Można również wydrukować arkusz na drukarce, co jednak nie jest sprawą prostą, gdyż prezentowana wersja "VisiCalc" jest przystosowana do współpracy z drukarką Atari 825. Gdy jej nie posiadamy, należy zawartość arkusza zapisać na dyskietce jako zbiór w kodzie ASCII (komenda/PF), po czym wydrukować go przy użyciu programu do przetwarzania tekstów (np.: "AtariWriter" czy "SpeedScript").

Podany przykład jest stosunkowo prosty, mimo że wykorzystuje najczęściej używane możliwości programu. Użytkownik może zmienić przykład przez jego rozbudowanie (np.: obliczenia dla całego roku) lub wprowadzenie nowych formuł i wartości ograniczających.

W praktyce gospodarczej "VisiCalc" powinien znaleźć zastosowanie do kalkulacji kosztów, planowania krótko- i średniookresowego czy nawet do wariantowego obliczania obciążeń na PFAZ.

ADAM STAWOWY

OZNACZENIA:

w – dowolne wyrażenie mające określoną wartość liczbową,

ciąg – kolejne komórki w wierszu lub kolumnie określone przez współrzędne pierwszej i ostatniej np. B1... B15,

lista – kombinacja wyrażeń i ciągów oddzielonych przecinkami.

FUNKCJE:

ABS(w) wartość bezwzględna z wyrażenia
AVERAGE(lista) wartość średnia wystąpień w liście
COUNT(lista) podaje liczbę niezerowych wystąpień w liście
EXP(w) podaje wartość e do potęgi argumentu
INT(w) wartość całkowita wyrażenia
LN(w) logarytm naturalny
LOG10(w) logarytm dziesiętny
LOOKUP(w, ciąg) podaje liczbę komórek, których wartości są mniejsze lub równe wartości wyrażenia
MAX(lista) określa wartość maksymalną w liście
MIN(lista) określa wartość minimalną w liście
NPW(w, ciąg) służy do obliczania procentu składanego w ciągu komórek (wartość procentu podana w wyrażeniu)
SQRT(w) pierwiastek kwadratowy
SUM(lista) suma wartości w liście

FUNKCJE TRYGONOMETRYCZNE (wartość kąta podana w radianach):

ACOS(w) arcuscosinus
ASIN(w) arcussinus
ATAN(w)arcustangens
COS(w) cosinus
SIN(w) sinus
TAN(w) tangens

KOMENDY:

/B zmazanie zawartości komórki wskazanej przez kursor
/C zmazanie całego arkusza
/D zmazanie wiersza (/DR) lub kolumny (/DC) wskazanych przez kursor
/FF formatowanie komórki wskazanej przez kursor:
/FG powrót do ogólnego formatu (liczba rzeczywista)
/FI wynik w postaci liczby całkowitej
/FL wynik justowany z lewej strony
/FR wynik justowany z prawej strony
/F$ wynik w postaci dolarów i centów
/FF graficzne przedstawienie wyniku jako "gwiazdki" (ich liczba odpowiada części całkowitej wyrażenia)
/G działanie na całym arkuszu lub okienku:
/GC wybór szerokości kolumny (3 do 39 znaków)
/GF formatowanie całego arkusza (opcje jak przy formatowaniu komórki)
/GO określenie kierunku rekalkulacji:
/GOC wzdłuż kolumn
/GOR wzdłuż wierszy
/GR sposób rekalkulacji:
/GRA automatycznie
/GRM przez użytkownika
/I wstawienie pustego wiersza (/IR) lub kolumny (/IC)
/M przesunięcie wiersza lub kolumny w inne miejsce arkusza
/PF zapisanie na dyskietce arkusza lub jego części jako zbioru w kodzie ASCII
/PP drukowanie arkusza lub jego części na drukarce
/R replikowanie (kopiowanie) zawartości komórki lub ciągu komórek w wybranym miejscu arkusza
/SD "zmazanie" na dyskietce zbioru o podanej nazwie
/SI formatowanie dyskietki
/SL wczytanie z dyskietki zbioru zawierającego wszystkie informacje o arkuszu
/SS zapisanie na dyskietce zbioru zawierającego wszystkie informacje o arkuszu
/SQ "zmazanie" programu "VisiCalc", powrót do systemu operacyjnego
/T umiejscowienie na trwałe na ekranie wierszy powyżej kursora (/TH), kolumn na lewo od kursora (/TV), wierszy powyżej i kolumn na lewo od kursora (/TB) (podczas "przewijania" ekranu wiersze (kolumny) te są stałe widoczne)
/W podział ekranu na dwa okienka wzdłuż wiersza (/WH) lub kolumny (/WV) wskazywanych przez kursor
/WS synchroniczne przewijanie okienek
/WV niezależne przewijanie okienek
/– wypełnienie komórki podanym znakiem
> skok do komórki o podanych współrzędnych

37 ◀

radzi sobie w tej sytuacji, pytając nas do jakiego pola (współrzędne) ma powtórzyć zawartość elementu źródłowego, na którą mogą oczywiście składać się konkretne wartości, jak i wyrażenia matematyczne, przy czym automatycznie uwzględniana jest ich względność. W przypadku jeżeli chcemy, aby wartość z jakiegoś pola brana była jako stała, to przed jego współrzędnymi należy podać "k" np: ka6.

Program "OMNICALC 2" posiada także takie udogodnienia jak np: automatyczne obliczanie sum (total) i sum cząstkowych (sub – total) wierszy i kolumn; jednokrotne i iteracyjne przeliczanie tabeli, w tym drugim przypadku obliczenia trwają do momentu aż wartość z prawego dolnego rogu tabeli osiągnie zero; ułatwienie przy wprowadzaniu nazw miesięcy (wystarczy podać skrót angielskiej nazwy początkowego miesiąca, a program uzupełni to nazwami do końca roku); dwie możliwości wyświetlania zawartości w postaci całkowitej i stałoprzecinkowej z dwoma miejscami po przecinku, przy czym program cały czas wszelkie wartości przechowuje z normalną dokładnością Spectrum.

W celu utrwalenia wykonanych obliczeń można je wydrukować na drukarce pełnoformatowej (z odpowiednim interfejsem) lub na ZX Printerze. Możliwe jest drukowanie całej tabeli lub pewnych jej fragmentów. Jeżeli zakres wydruku przekracza szerokość drukowania drukarki, to program automatycznie przenosi wydruk nie mieszczących się kolumn na nową stronę. W przypadku gdy mamy podłączoną drukarkę graficzną, możemy skopiować na niej wykres słupkowy (histogram).

"OMNICALC 2" jest przygotowany do współpracy z INTERFACE 1 Sinclaira. Po wgraniu do komputera program bada czy podłączone są stacje microdrive'ów, jeżeli tak – daje użytkownikowi możliwość przegrania na kasetkę microdrive czekając około 10 sekund na naciśnięcie dowolnego klawisza, aby nagrać się pod nazwą RUN (program pod taką nazwą ładuje się automatycznie bez kłopotliwego pisania pełnej komendy LOAD ✻ "m";1;"nazwa"). Wykonywanie niektórych rozkazów INTERFACE 1 ułatwiają specjalne komendy dodatkowe (CAT, ERASE, OPEN#3, CLOSE#3).

TOMASZ ZIELIŃSKI

FUNKCJE:

5,6,7,8	**– przesuwanie kursora po oknie;**
CS+5,6,7,8	**– przesuwanie okna po arkuszu tabeli;**
CS+9	**– rysowanie wykresu słupkowego;**
A (amend)	**– dodawanie lub kasowanie wierszy lub kolumn;**
C (calculate)	**– przeliczanie tabeli (jednokrotne lub iteracyjne);**
D (decode)	**– dekodowanie zawartości pola z kursorem;**
G (go to)	**– przemieszczanie kursora po danej pozycji oraz okna zawierającego kursor;**
I (input)	**– wprowadzanie informacji w pole zawierające kursor;**
K (copy)	**– kopiowanie zawartości pola z kursorem do innego miejsca;**
L (load)	**– ładowanie nagranej tabeli lub obszaru roboczego do komputera bądź inicjowanie nowej tabeli;**
M (mode)	**– przejście do wyświetlania z dwoma miejscami po przecinku;**
P (print)	**– wydruk tabeli na drukarce;**
Q (quick)	**– zwiększenie szybkości przesuwania okna po arkuszu tabeli poprzez rezygnację z wyświetlania wartości pól;**
R (repeat)	**– powtarzanie zawartości pola z kursorem w kolumnie lub wierszu do podanej współrzędnej;**
S (save)	**– nagrywanie tabeli lub obszaru roboczego na magnetofonie lub microdrive;**
T (total)	**– sumowanie kolumny bądź wiersza od miejsca z poprzednią sumą cząstkową (sub – total) lub od początku do pola z kursorem;**
W (workspace)	**– przesyłanie wartości maksimum 60 pól do obszaru roboczego w celu wykonania histogramu, przeniesienia w inne miejsce arkusza lub do innej tabeli;**
X (extended comands)	**– komendy dodatkowe umożliwiające wykonanie pewnych rozkazów microdrive'ów: Open 3 – przydziela kanał RS 232 dla drukarki, Drop3 – zamyka kanał otwarty opcją powyżej, Cat – wyświetla katalog kasetki microdrive, Erase – kasuje zbiór na kasetce, Baud – ustala prędkość transmisji dla RS 232, G–ad – określa adres programu drukującego grafikę, używanego do skopiowania histogramu;**
Y (year)	**– szybkie wprowadzenie skrótów nazw miesięcy od pola z kursorem i nazwą początkowego miesiąca do końca roku.**

Zmienne w programowaniu

"Programowanie jest sztuką konstruktywną. W jaki sposób uczyć konstruktywnej i twórczej działalności? Jedną z metod jest wyabstrahowanie, na podstawie wielu przykładów, zespołu elementarnych zasad tworzenia i przekazanie ich w systematyczny sposób. Jednakże programowanie jest dziedziną wielce zróżnicowaną. Przypuszczenie, że kiedykolwiek można by jego naukę skondensować w postaci ścisłych recept, wydaje się niesłuszne..."

N. Wirth

JĘZYKI KLASYCZNE

Pod hasłem "języki klasyczne" będziemy rozumieli języki programowania typu Basic, Fortran, Pascal. Jednym z kryteriów podziału języków jest bowiem filozofia rozumienia zmiennej, a co za tym idzie i programowania. Dzieli się więc języki na tzw. obiektowo zorientowane i klasyczne, czyli "Basicofortranopascalopodobne".

Każdy kto napisał w swoim życiu choćby jeden prosty program, zauważył na pewno, że oprócz algorytmu – czyli "przepisu" na rozwiązanie zadania, trzeba rów-

▶ 40

39 ◀

nież zadbać o "pamiętanie" – czyli przechowywanie w pamięci komputera – pewnych pośrednich wyników, informacji, danych. Weźmy banalny przykład:

```
10 PRINT "Ile masz lat?"
20 INPUT wiek
30 PRINT "O! Masz już ";wiek;" lat!"
```

Nic tu się mądrego nie dzieje, ale spróbujmy posadzić przy klawiaturze osobę, która pierwszy raz w życiu widzi komputer i nic o nim nie wie. Będzie najczęściej bardzo zaskoczona: jaki ten komputer "mądry" – on wie, ile ja mam lat! Ta "mądrość" komputera dla nas jest sprawą oczywistą – zmienna wiek. Dla osoby nic o informatyce ne wiedzącej wielkim zaskoczeniem jest fakt, że komputer może pamiętać. Sposób, w jaki program pamięta wyniki, informacje – dane, a raczej konstrukcje języka programowania, których do tego celu używa, informatycy nazywają strukturami danych.

Wybór struktur danych jest równie ważny jak wybór algorytmu rozwiązania. O tym czy program jest dobry, a więc skuteczny, niezawodny, przejrzysty (bardzo ważna cecha dobrego programu), optymalny, obok algorytmu decydują również struktury danych. W najprostszych przykładach jest oczywiste, jakich struktur użyć. Jeśli mamy np. posortować (uporządkować – ustawić „po kolei") 50 liczb, to wiadomo, że użyjemy tablicy o 50 elementach, a nie 50 różnych zmiennych liczbowych. Wyobraźmy sobie algortym sortujący, który używa tylko pojedynczych zmiennych!

Nie zawsze jednak sprawa jest tak oczywista. W skomplikowanych programach bardzo długo trwają (czasem w ogóle najdłużej) przygotowania projektów struktur danych. Duże systemy baz danych są zwykle oparte na bardzo wyrafinowanych sposobach pamiętania danych i dostępu do nich. Tutaj bez dobrego przemyślenia problemu nie można rokować nadziei na to, że program, a raczej system programów, będzie spełniał oczekiwania przyszłych użytkowników. Nie przez przypadek Niklaus Wirth zatytułował swoją wspaniałą książkę: "Algorytmy+Struktury danych=Programy".

Ci, którzy programują tylko w Basicu, nie mają zbyt dużego wyboru, jeśli chodzi o konstrukcję języka służącego do „pamiętania"

Są tu do dyspozycji tylko zmienne i tablice (zmienne indeksowane). Teoretycznie za pomocą takich struktur można zaprogramować wszystko, ale nakład pracy jest czasem niewspółmierny do otrzymanych wyników (często też znalezienie błędu w skomplikowanym programie pisanym "na siłę" w Basicu czy Fortranie jest praktycznie niemożliwe). Zmienne i tablice są narzędziami bardzo prostymi, ale jednocześnie mało uniwersalnymi i niewygodnymi w większości zastosowań. Niech pracują za nas translatory – tak powiadają programujący w językach wyższych poziomów.

Fortran nie wnosi nic nowego do struktur danych w porównaniu z Basicem

Tu również podstawową konstrukcją jest zmienna prosta i tablica. Programy pisane w Fortranie są wprawdzie bardziej przejrzyste, pojawia się podprogram – procedura lub funkcja – nazwany i mogący posiadać parametry. Nadal jednak umiejętność programowania polega (oprócz dzielenia problemu na podproblemy) na zręcznym manipulowaniu zawartością pamięci, która jest reprezentowana przez zmienne proste i indeksowane. Ponieważ najczęściej Fortran nie kontroluje odwołań do tablic, tzn. można np. użyć elementu A/40/ tablicy zadeklarowanej jako 30–elementowej, zręczny programista, znający jeszcze dobrze działania translatora, może ingerować z poziomu Fortranu w kod maszynowy. Z drugiej strony programista niedoświadczony ma wiele kłopotu, by zlokalizować błąd powstały przez przypadkowy dostęp do zmiennej indeksowanej.

Co najmniej o klasę wyżej stoi Pascal

W tym języku istnieje możliwość deklaracji typu. Innymi słowy, można wymyślić sobie dowolną strukturę opartą na wcześniej zadeklarowanych typach i typach prostych (liczby, znaki, tablice itp.). Dochodzi tu pojęcie REKORDU oraz zmiennych wskazywanych i wskaźnikowych. Rekord jest typem złożonym, który doskonale reprezentuje obiekty opisane przez kilka (lub bardzo dużo) różnych informacji. Jeśli zmienna jest typu rekordowego, to ma najczęściej kilka tzw. pól, w których zawarte są kolejno różne informacje o danym obiekcie. Upraszczając nieco sprawę, zmienna rekordowa to tak jakby "pęczek" zmiennych spiętych razem w jedną zmienną złożoną. Na przykład:

```
type człow = record  płeć : char;
                     wiek : integer;
                     wzrost : integer
             end;
var człowiek : człow;
```

Zmienna "człowiek" jest typu rekordowego (rekord człow) i może zawierać trzy informacje – znak określający płeć, dwie liczby całkowite oznaczające wiek i wzrost. Odwołanie do konkretnej już informacji odbywa się przez podanie nazwy zmiennej rekordowej, kropki i nazwy pola np.: "człowiek.wzrost". Konkretny przykład instrukcji Pascala to:

```
...
if człowiek.płec = "K" then writeln
człowiek.wiek : 2); ...
("wiek tej pani wynosi "
```

Instrukcja oznacza, że jeśli zmienna człowiek opisuje kobietę, to należy wydrukować (wyświetlić) tekst "wiek tej pani wynosi" oraz wartość pola "wiek" na dwóch miejscach (w dwóch kolumnach).

Pascal wprowadza również pojęcie wskaźników i zmiennych wskazywanych. Najogólniej mówiąc, zmiennej wskazywanej nie ma w chwili wywołania programu i choć jest deklaracja odpowiedniej struktury, nie istnieje odpowiadający jej obszar pamięci komputera. Zmienna powstaje dopiero po wykonaniu instrukcji NEW skojarzonej z odpowiednią zmienną wskaźnikową (wskaźnikiem). Najczęściej zmienne wskazywane stosuje się, gdy wiadomo z góry, ile pamięci zarezerwować na dane. Realizowany jest tutaj bowiem tzw. dynamiczny przydział pamięci. Zmienna wskazywana jest najczęściej rekordem, którego jednym z pól jest wskaźnik – zmienna wskaźnikowa mogąca pokazywać na podobną zmienną rekordową. Dlatego też zmienne tego typu używane są przy algorytmach operujących na listach lub bardziej złożonych strukturach listowych – drzewach, grafach.

W Pascalu pojawia się również pojęcie typu zbiorowego oraz plikowego jako typów strukturalnych znanych od początku, tzn. nie wymagających deklaracji. Nie będę się rozwodził na ten temat, wystarczy intuicyjna znajomość pojęcia zbioru (matematyka – szkoła podstawowa) oraz pliku – pojęcia na pewno nieobcego interesującym się informatyką.

Bardzo istotną zaletą Pascala jest możliwość użycia rekursji, bardzo silnego narzędzia programistycznego, co pozwala na stosowanie eleganckich, przejrzystych algorytmów w wielu problemach. Być może kilkanaście ostatnich zdań było zbyt zawiłych, wystarczy jednak zapamiętać, że w Pascalu zmienne mogą być nie tylko proste i tablicowe, ale mogą reprezentować przeróżne, często bardzo skomplikowane, struktury.

Jednak zarówno w Basicu i Fortranie, jak i w Pascalu, filozofia rozumienia jest podobna. W każdym z tych języków mamy do czynienia z deklaracją zmiennej (jawną lub ukrytą). I tak w Pascalu każda nawet najprostsza zmienna musi być zadeklarowana np.:

```
var x : integer;
    z : array (.1..20.) of char;
```

W Fortranie dobra zasada nakazuje deklarować zmienne, lecz tutaj jeśli używamy nowej nazwy (identyfikatora) – niewyspecyfikowanej wcześniej – błędu nie ma. Typ takiej zmiennej zależy wtedy od pierwszej litery nazwy i jeśli jest to: i, j, k, l, m lub n, zmienna jest całkowita (może przybierać tylko wartości liczbowe niеułamkowe), w innym przypadku rzeczywista (dowolne liczby). Tablice podobnie jak w Pascalu muszą być deklarowane, ale można podać tylko wymiar (zakres) – typ elementów może określać pierwsza litera.

W Basicu zmiennych prostych ne deklaruje się. Typ zmiennej użytej po raz pierwszy ustala się tak: jeśli drugim znakiem nazwy jest "$" lub "£" (ten drugi znak jest dozwolony w niektórych realizacjach Basicu), zmienna jest napisowa, co znaczy, może przechowywać napisy, czyli ciągi znaków (łańcuchy). W innym przypadku zmienna jest liczbowa bez rozróżniania na całkowitą czy rzeczywistą. Tablice deklaruje się podając nazwę i wymiar. Typ elementów tablicy ustala jej nazwa, na zasadach identycznych jak dla zmiennych prostych. W wielu realizacjach Basicu nazwy zmiennych mogą być najwyżej dwuliterowe, co oczywiście bardzo utrudnia użycie nazw znaczących (mnemotechnicznych).

Ponieważ w Basicu i Fortranie nie ma obowiązku deklaracji zmiennych prostych, często dla wygody lokalnie wprowadzamy sobie jakąś nową zmienną, czy to na użytek pętli FOR, czy na chwilowe przechowanie jakiejś wartości. Jest to z pewnością bardzo wygodne, lecz wymaga rozwagi. Olbrzymia liczba błędów spowodowana jest spontanicznym użyciem dowolnych nazw – często jednocześnie tej samej do różnych celów – zwłaszcza w dużych programach. Dlatego też Pascal (ale nie tylko z tego powodu) wymusza na nas zadeklarowanie każdej najprostszej zmiennej i sam również kontroluje sposób jej użycia. Nie można odwołać się poza zakres tablicy lub wstawić liczby rzeczywistej w zmienną całkowitą. Każda zmienna powinna być użyta zgodnie ze swoim przeznaczeniem (deklaracją) – tak, oczywiście, powinno być w idealnym programie.

Dotyczy to również dobieranych przez nas nazw zmiennych. Tutaj pole do popisu programujący mają prawie w każdym języku. Każda nazwa (identyfikator) powinna kojarzyć się ze swoim przeznaczeniem – jak mówią fachowcy "powinna być mnemotechniczna". Wiadomo, zmienna "wiek" powinna przechowywać czyjś tam wiek, a nie numer butów... Jeśli zaś zmienna "i" lub "x" jest używana do czegoś innego niż w pętli FOR i=... dobrze jest opatrzyć ten fragment programu komentarzem.

W prostych i krótkich programach da się jeszcze kontrolować użycie każdej zmiennej, lecz niech ktoś spróbuje uruchomić duży (i niebanalny) program bez dobrego nazewnictwa! Duży program, to w moim rozumieniu kilka tysięcy linii, lecz trudno mierzyć program na metry wydruku – ważne jest, oczywiście, co się tam dzieje w środku (czy jest to drukowanie pięknego kolorowego kalendarza ze szlaczkami, czy program grający w szachy lub brydża). Po uruchomieniu programu można pozmieniać sobie nazwy, choćby po to, by zmniejszyć pamięć przydzielaną na tekst lub by uczynić go niezrozumiałym dla ewentualnych włamywaczy, ale warto też zostawić sobie oryginał (rzadko się zdarza, że program nie ma błędów).

Zastanówmy się jednak bardziej ogólnie, czym jest zmienna w językach klasycznych. Jak wiemy, na zmiennej można pamiętać pewne wartości, czy to liczbowe, czy napisowe, czy też bardziej złożone w Pascalu. Jest to zatem obszar pamięci komputera, w którym przechowywane są, lub mogą być przechowywane, pewne informacje. Obszar pamięci, który odpowiada danej zmiennej, ustala translator języka. I tu uwaga! – w językach klasycznych jest to ściśle określony obszar o konkretnej, stałej wielkości. Translator ustala "pudełeczko", dla zmiennej liczbowej małe, dla napisowej większe, a dla tablicy np. DIM A(30) pudełko, w którym mieści się 30 pudełeczek elementarnych. Pudełko jest przydzielane zmiennej w chwili deklaracji lub w chwili pierwszego wystąpienia (Basic, Fortran). Ponieważ w Pascalu można deklarować zmienne lokalne procedur i funkcji – pudełka im przeznaczone tworzone są dopiero w chwili wywołania danego podprogramu i znikają po zakończeniu jego działania. Jest to ważna cecha odróżniająca Pascal od Basicu i Fortranu, bo niektóre zmienne (lokalne) "dostają" pudełka pamięci dopiero w czasie działania programu, a nie w momencie translacji jak zmienne globalne – deklarowane w programie głównym, czy wszystkie zmienne Basica lub Fortranu. Zauważmy, że bez względu na to, kiedy pudełko zostało utworzone, jeśli już jest, można schować w nim tylko tyle, ile się mieści (pewną określoną porcję) – nie jest ono z gumy.

Podsumujmy krótko powyższe informacje. Programując, musimy używać struktur danych. W językach prostych, ale zarazem prymitywnych, używamy zmiennych prostych i indeksowanych (tablic). Dla każdej zmiennej utworzone jest "pudełeczko" pamięci o stałej wielkości. Pudełko to część pamięci komputera, która jest na nie przeznaczona aż do chwili zakończenia programu lub podprogramu w Pascalu.*)

Ponieważ pamięć komputera nie jest nieskończona, ale jednocześnie nie taka znów mała, warto dbać o rozsądne gospodarowanie "pudełkami". Nikt przecież nie kupuje regału, by trzymać tam jedną lub dwie książki, ale również nie używa tego samego słoika do przechowywania kompotu i śledzi. Ten zbyt może trywialny przykład miał uzmysłowić, że nie należy przesadzać w każdą stronę. Nie powinno się być rozrzutnym, ale też i zbyt skąpym. Wspomniana wcześniej zmienna "wiek" użyta do czegoś innego może wprowadzić chaos i program stanie się mało czytelny i trudny do poprawienia.

Programowanie w językach klasycznych polega więc na umiejętnym operowaniu zawartością pudełek, szufladek, przegródek. Wyobraźmy sobie dobrze poinstruowanego szybkiego służącego, który znajduje się w pokoju pełnym takich szufladek. Wyjmuje on przeróżne rzeczy z tych schowków, przekłada, waży, porównuje, dodaje, znów chowa, i tak dalej. Gdy skończy mu się instrukcja, ustawia na tacy zawartość szuflad wyspecyfikowanych na końcu i niesie swojemu panu, czyli nam. To co on nam przyniesie (i czy w ogóle przyniesie) zależy od tego co ma w instrukcji – od naszego programu.

Musimy więc dobrze wiedzieć, jakie meble stoją w pokoju, do którego wysyłamy służącego, by nie kazać mu wstawić telewizora do lodówki. Zmienne w omawianych wyżej językach mają ściśle określony typ i swoją reprezentację w postaci obszaru pamięci komputera. Programowanie to umiejętne operowanie tą pamięcią, a w Pascalu można robić w przejrzysty i bardzo elegancki sposób. Podobną elegancję można osiągnąć i w językach niższych poziomów, ale jest to znacznie trudniejsze do zrealizowania.

JĘZYKI OBIEKTOWO ZORIENTOWANE

Tym mianem określa się języki programowania, w których wszystko, bez względu na znaczenie, jest obiektem. Podobnie jak liczba jest obiektem, jest nim również zbiór liczb, czy w ogóle obiektów, zorganizowany w listę i jest nim także treść procedury. Do najbardziej znanych języków obiektowo zorientowanych należą Lisp, Prolog i Logo. Ponieważ Logo jest w zasadzie dialektem Lispu i to skierowanym na prostotę programowania (lecz nic nie tracącym na sile Lispu) można prezentować te dwa języki razem pod postacią Logo. Odmian Prologu jest też niemało. Do niedawna dość dobrze znany w Polsce był tzw. ODRA PROLOG – nazwany tak ze względu na realizację na komputerach serii ODRA. Obecnie w dobie mikrokomputerów dość powszechny stał się Micro Prolog z realizacją na wszystkich chyba typach mikrokomputerów.

Pierwszy problem pojawia się na samym początku, a mianowicie pojęcie programu. W Lispie, a zatem i w Logo oraz w Prologu, to pojęcie właściwie nie istnieje. Program jako całość, konkretny tekst napisany od nagłówka do słowa "end" czy "stop", wywoływany najczęściej przez podanie nazwy lub RUN i ewentualnie etykiety (na ZX Spectrum), nie ma zupełnie swojego odpowiednika w omawianych językach. Języki te są bowiem proceduralne i programowanie w nich to pisanie małych (najczęściej) procedur rozwiązujących konkretne drobne rzeczy. W Prologu procedury te nazywa się klauzulami, gdyż w rzeczywistości nie są to "programiki", lecz definicje obiektów. Zwykle procedury te lub klauzule używają innych, te z kolei innych i w rzeczywistości może być to bardzo rozbudowana struktura. Wywołanie głównej procedury (niech zwolennicy Prologu nie obrażą się za tę nieścisłość) to jakby wywołanie programu, ale można też wywoływać osobno każdą procedurę i na przykład testować sobie jej poprawność.

Tak więc w środowisku Logo czy Prologu mamy do dyspozycji pojedyncze procedury, które wiążemy w logiczną całość lub używamy ich samodzielnie, w zależności od tego co chcemy osiągnąć

Procedury te mają najczęściej parametry. Owe parametry nie są jednak reprezentantami określonych ściśle obiektów. W chwili wywołania w ich miejsce można wstawić dowolny obiekt – parametr bowiem określa, że tutaj ma być "coś", a nie konkretnie liczba, napis, zmienna określonego typu itp. W Prologu parametr to zmienna, o której nic nie wiemy. W chwili wywołania badane jest przez analizę dotychczas zdefiniowanych obiektów, co może być tą zmienną i jeśli coś może, pojawią się odpowiedź. Na tej analizie głównie polega działanie programów prologowych i nazywa się to UZGADNIANIEM.

Nie jest moim celem przedstawianie sposobów programowania w Prologu, Lispie czy Logo – potrzeba na to kursów programowania, chciałbym jedynie pokazać różnicę w podejściu do zmiennej w językach obiektowo zorientowanych w stosunku do języków klasycznych. Tak więc parametr jest szczególnym rodzajem zmiennej (podobnie w językach klasycznych). Określa on pewną wartość, jaka będzie wstawiona w jego miejsce w chwili wywołania procedury i jako taki nie przedstawia żadnej. Ma znaczenie napisu, mówiącego, że w danym miejscu w danej instrukcji będzie kiedyś konkretna wartość. Dlatego niektórzy uważają, że parametr nie jest zmienną sensu stricto. W językach klasycznych gdzie istnieje pojęcie procedury (np. Fortran, Pascal) parametr musi mieć ściśle określony typ.

W Logo czy Prologu nic takiego nie występuje, parametr określa dowolny obiekt i przy wywołaniu można zamiast niego wstawić zarówno liczbę jak i słowo (napis)

Oczywiście jeśli będziemy potem próbować dzielić parametr np. przez 2, w drugim przypadku powstanie błąd, ale jeśli będziemy drukować go "na wyjście", to procedura w każdym przypadku zadziała poprawnie.

Zastanówmy się nieco dłużej nad instrukcją przypisania w Logo. Jest ona pewnym odpowiednikiem instrukcji podstawienia (nazywanej też często instrukcją przypisania) w językach klasycznych, a więc "LET" w Basicu, "=" w Fortranie i ":=" w Pascalu. Jest to "MAKE" w Logo, a w polskim Logo "PRZYPISZ" lub w skrócie "PRZYP." W językach klasycznych instrukcja podstawienia to schowanie do pudełka odpowiadającego zmiennej stojącej po lewej stronie, wartości odpowiadającej prawej stronie. Może być to wartość podana dosłownie (stała) np LET i =1 lub wzięta z innego pudełka np. LET i =k. Zawsze jednak wartość ta musi zmieścić się w pudełku odpowiadającym lewej stronie. Instrukcja podstawienia jest zatem przypisaniem wartości do nazwy (a raczej pudełka reprezentującego nazwę). W Logo natomiast jest dokładnie odwrotnie. Następuje tutaj przypisanie nazwy do wartości. I tak dozwolone jest: PRZYPISZ "człek 22 jak również później: PRZYPISZ "człek [kobieta 165 25]. Nazwa stojąca po lewej stronie określa zmienną, ale zmienną ta to nie pudełko, ale raczej karteczka-etykietka z nazwą, która może być przyczepiona do dowolnego obiektu. Dlatego trudno w Logo mówić o typie zmiennej, bo przecież karteczka to tylko nazwa – mówimy o pojemności butelki, a nie o pojemności naklejki na niej. Jeśli ta etykietka jest do czegoś przyczepiona, poprzez nazwę etykietki możemy używać obiektu pod etykietką. Jeśli jednak ponownie wykonamy instrukcję PRZYPISZ z tą samą nazwą, karteczka z tą nazwą zostanie "przepięta" na nowy obiekt. Tracimy jednocześnie dostęp do starego obiektu.

▶ 42

PC-DOS

O systemie PC-DOS nie można mówić w oderwaniu od komputera, dla którego został zaprojektowany, tzn. IBM PC. Pierwotnie maszyna ta była produktem mającym obronić pozycję rynkową koncernu IBM, dotychczas specjalizującego się w dużych systemach komputerowych. Mikrokomputery stawały się coraz bardziej popularne i powszechne, więc IBM nie mógł dopuścić, aby grupę jego stałych klientów przechwyciły inne firmy. Było to równocześnie eksperymentalne wejście na dopiero tworzący się rynek 16-bitowych mikrokomputerów.

IBM PC wprowadzono do sprzedaży we wrześniu 1981 r. i stał się prawie natychmiast ogromnym sukcesem rynkowym, ustanawiając standard na mikrokomputery 16-bitowe. Sukces ten był rezultatem zachwytu i entuzjazmu użytkowników tego modelu oraz sławy i poważania, jakimi cieszył się producent.

W ciągu zaledwie 2 lat użytkowania IBM:

- Opanował 26% rynku mikrokomputerowego wyprzedzając wszystkich producentów małych systemów.
- Ustanowił rodzinę mikroprocesorów INTEL 8088/8086 jako przemysłowy standard dla mikrokomputerów.
- Spowodował ogromny wzrost zainteresowania mikrokomputerami (obroty w tej dziedzinie wzrosły trzykrotnie).
- Zrewolucjonizował spojrzenie IBM i innych czołowych firm na małe systemy. Zaczęły mieć one dla nich ogromne, wręcz strategiczne znaczenie.
- Zmienił spojrzenie wielu użytkowników na komputery osobiste. Z zabawki stały się bardzo ważnym elementem w systemach informacyjnych.

Ten sukces mikrokomputera to równocześnie sukces jego systemu operacyjnego. PC-DOS stał się standardem wśród 16-bitowych mikrokomputerowych systemów operacyjnych.

BUDOWA SYSTEMU

System operacyjny PC-DOS zbudowany jest z następujących modułów:

1. Z rekordu wprowadzającego (ang. boot record) rezydującego na ścieżce 0 (sektor 1, strona 0) na każdym dysku sformatowanym poleceniem FORMAT. Przy próbie załadowania DOS z niesystemowej dyskietki rekord ten powoduje pojawienie się odpowiedniego komunikatu na konsoli.
2. Z modułu BIOS umieszczonego w pamięci ROM, zawierającego podstawowe procedury obsługi urządzeń zewnętrznych.
3. Z pliku IBMBIO.COM umożliwiającego połączenie systemu z procedurami ROMBIOS.
4. Z właściwego programu DOS (plik IBMDOS.COM) zapewniającego współpracę na wysokim poziomie z programami użytkowymi(odpowiednik BDOS w CP/M).
5. Z procesora poleceń COMMAND.COM zapewniającego odpowiednią interpretację poleceń systemowych. Procesor ten składa się z kolei z następujących modułów:

---

41 ▶

Ciekawy jest przykład instrukcji: PRZYPISZ "X "X. Zmiennej, czyli nazwie X, przypisujemy wartość napisu X. Znaczy to, że karteczka z nazwą X zostaje przyczepiona do obiektu będącego napisem X. Ale w Logo ten sam obiekt jest przecież raz reprezentowany – jeden jest tylko napis X. Taką sytuację można wyobrazić sobie jako złożenie karteczki na pół. Gdy chcemy sprawdzić, co jest pod karteczką, patrzymy pod nią, a tam ona sama... Można też sprawdzić: co jest wartością obiektu, którego nazwa jest wartością obiektu, którego nazwa... ...nazwą jest X. Oczywiście wynikiem będzie napis X. Poprzewracamy sobie karteczkę z jednej strony na drugą widząc zawsze napis X.

Podsumowując: w językach obiektowo zorientowanych pojęcie zmiennej jest więc inne niż w językach klasycznych. Mamy do czynienia z nazwami, które mogą zmieniać miejsce swojego "przyczepienia" i oznaczać różne jakościowo obiekty, zarówno proste liczby jak i złożone struktury. Nie istnieje pojęcie typu zmiennej. Zmienna albo nie ma wartości (nie jest przyczepiona do niczego), albo reprezentuje obiekt – coś dowolnego pod względem znaczenia. Programowanie z użyciem zmiennych, to jakby poruszanie się we wszechświecie dowolnych obiektów, które sobie "pływają" w bezładny sposób, a my mając jakiś obiekt na "widelcu" możemy przyczepić mu etykietkę z nazwą i tym samym uczynić go dostępnym w każdej chwili. Mając dostępne dwa obiekty możemy je złączyć w jeden i też nazwać, możemy również obiekt złożony dowolnie podzielić.

To zupełnie odmienne podejście do zmiennej i programowania ma jeszcze wielu przeciwników. Piszę "jeszcze", bo liczba ich stale maleje. Lisp i Prolog są już znane od wielu lat – Lisp co najmniej od 25-ciu. Ich popularność stale rośnie, zwłaszcza Logo i Prologu. Na początku były to języki tylko dla wtajemniczonych, a jeszcze w roku 1972, więc 12 lat po powstaniu Lispu, E. W. Dijkstra (postać znana w całym informatycznym świecie) pisał, że Lisp to "najbardziej wyrafinowany sposób nieprawidłowego wykorzystywania komputera".

Zbytni formalizm tego języka czyni go niedostępnym dla wielu – lecz popularność Logo, filozoficznie mało różniącego się, z powodzeniem to rekompensuje. O Prologu mówi się, że jest językiem przyszłości i choć różni się zasadniczo od Lispu i Logo, to należy do tej samej kategorii. Na pewno wszystkie te języki są bliższe sobie niż każdy z nich osobno Pascalowi. Warto chyba, by ze względu na przepowiednie o komputerach piątej generacji, zainteresować się tymi językami. Sprawa jest dość prosta – na ZX Spectrum jest przecież i Micro Prolog, i Logo, nawet w polskiej wersji. Niedługo mają też ukazać się na rynku polskie podręczniki Logo i Micro Prologu, co na pewno znacznie ułatwi życie uczącym się tych języków.

ZBIGNIEW KASPRZYCKI

---

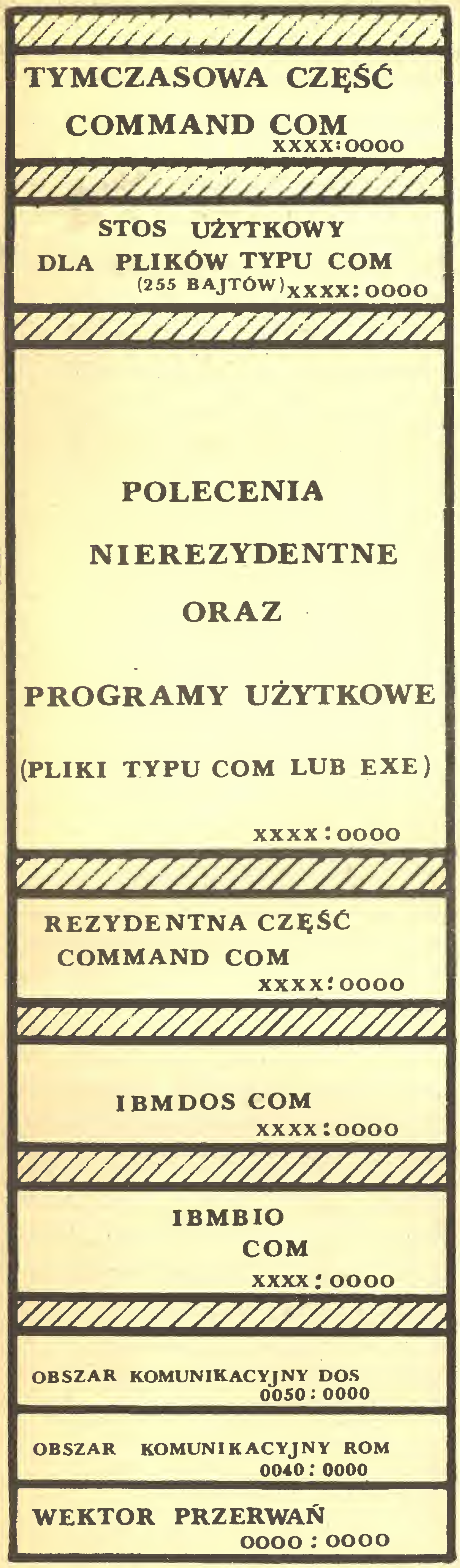
*) Pominąłem tu tzw. struktury dynamiczne – zmienne wskazywane w Pascalu. Pudełka pamięci tych zmiennych można samemu tworzyć lub zwalniać – instrukcje NEW i DISPOSE oraz RELEASE – w trakcie działania programu. Na nasz lokalny użytek nie jest istotne, jaki jest "życiorys" pudełka, tylko fakt, że kiedyś tam się pojawia.

W LOGO preferować należy programowanie bez użycia instrukcji PRZYPISZ (make). Zaleca się stosowanie parametrów i tylko w ostateczności nazw do pamiętania informacji. Dane, w większości przypadków, można przekazywać w programie – zespole procedur – za pomocą jedynie parametrów (a więc zmiennych szczególnego rodzaju). Oto przykład sortowania ciągu liczb bez użycia zmiennej:

```
?
OTO WSTAW :el :lis
JEŚLI PUSTE? :lis [wynik zdanie
:el :lis]
JEŚLI :el < PIERW :lis [wynik zd
anie :el :lis]
WYNIK ZDANIE PIERW :lis WSTAW :e
l BEZPIERW :lis
JUż

OTO SORT :lis
JEŚLI PUSTE? :lis [wynik []]
WYNIK WSTAW PIERW :lis SORT BEZP
IERW :liś
JUż
```

Wywołanie może być takie: PISZ SORT CZYTAJ LISTĘ. Liczby jakie wprowadzimy z klawiatury aż do wciśnięcia ENTER – czyli lista złożona z tych liczb, będzie wstawiona w miejsce parametru :lis procedury SORT. Co dalej stanie się z tą listą – to jest pytanie!



● Z procedur rezydentnych umieszczonych w pamięci zaraz za plikiem IBMDOS.COM. Są to procedury obsługi przerwań i standardowych błędów DOS oraz procedura ładowania innych nierezydentnych modułów.

● Z modułu inicjacyjnego umieszczonego w pamięci zaraz za procedurami rezydentnymi. Tu przekazywane jest sterowanie podczas inicjacji systemu. Zadaniem tego modułu jest wykonanie plików CONFIG.SYS oraz AUTOEXEC.BAT, jeżeli takie istnieją (CONFIG.SYS ustala inną od standardowej konfigurację systemu, AUTOEXEC.BAT – niestandardowe opcjonalne parametry pracy systemu). Następnie moduł ten określa adres, pod który można ładować programy. Pierwszy załadowany przez procesor poleceń program zajmuje obszar pamięci wykorzystany uprzednio przez ten moduł.

● Z modułu tymczasowego, ładowanego pod górne adresy pamięci. Jest to procesor poleceń sam w sobie, zawierający wszystkie wewnętrzne procesy poleceń (dodatkowe procesory stworzone przez użytkowników) oraz procedurę ładowania i wykonywania poleceń nierezydentnych (pliki o rozszerzeniu .COM lub .EXE).

Rozmieszczenie wszystkich tych modułów w pamięci operacyjnej ukazane jest na rysunku.

ZARZĄDZANIE PAMIĘCIĄ

PC-DOS zezwala na podział pamięci na pewne ciągłe fragmenty. Są one zorganizowane w łańcuch obszarów wolnych (tzn. nie przydzielonych żadnemu programowi, gotowych do wykorzystania) oraz łańcuchy obszarów przydzielonych poszczególnym programom. Dla każdego z nich DOS buduje blok kontrolny, w którym zapisane są pewne informacje organizacyjne, takie jak wielkość tego obszaru oraz adres następnego bloku kontrolnego w łańcuchu.

Taka organizacja pamięci sprawia, że operacje przydziału lub zwalniania pamięci sprowadzają się do prostych operacji na blokach kontrolnych. W dodatku uniknięto dość istotnego w CP/M-86 ograniczenia na ilość możliwych do opisania regionów pamięci.

ORGANIZACJA KATALOGU

Pierwsze wersje systemu PC-DOS używały jedynie prostej struktury katalogowej, takiej samej jak w CP/M-86 i innych systemach tej rodziny. Każda dyskietka zawierała jednopoziomowy katalog, zwany też skorowidzem (ang. directory), w którym mogły być zapisane 64 (lub 112 przy dwustronnej dyskietce) pliki.

Po dodaniu możliwości operowania na dyskach stałych, dużo bardziej pojemnych niż dyskietki, taka organizacja stała się niewystarczająca. Na pojedynczym dysku stałym mogą być zapisane tysiące plików. Przy prostej strukturze katalogowej dostęp do nich byłby bardzo wydłużony.

Począwszy od wersji 2.0, DOS daje możliwość lepszej organizacji dysku. Pliki można grupować i dla każdej takiej grupy zakładać ich własny skorowidz. Na tym samym dysku może być ich wiele, zorganizowanych w strukturę drzewiastą.

Przy formatowaniu dysku zakładany jest skorowidz zwany korzeniem lub katalogiem systemowym. Można w nim zapisać 64 lub 112 pozycji. Każda z nich może być nazwą pliku lub nazwą skorowidza. W nim z kolei każdą pozycją może być również nazwa pliku lub nazwa katalogu. W ten sposób tworzona jest prosta struktura drzewiasta znacznie skracająca czas dostępu do zbiorów.

ORGANIZACJA URZĄDZEŃ ZEWNĘTRZNYCH

PC-DOS rozróżnia dwa typy urządzeń zewnętrznych: blokowe (transmisje odbywają się całymi blokami znaków) tzn. stacje dysków stałych i elastycznych oraz znakowe (transmisje odbywają się znak po znaku), takie jak drukarka, konsola, czytnik i perforator taśmy papierowej.

Urządzenia blokowe są identyfikowane w systemie poprzez litery (A,B,C...) Ich liczba jest ograniczona. Limit wynosi 63. Pojedyncze urządzenie może obsługiwać kilka stacji dyskowych. W czasie transmisji blokowych system nie może wykonywać żadnych innych zadań.

Natomiast urządzenia znakowe posiadają swoje nazwy własne (LPT, CON, TR, TP). Transmisje przeprowadzane są poprzez specjalny kanał przy minimalnym zaangażowaniu procesora. W tym czasie system może rozwiązywać inne zadania, znacznie oszczędzając czas użytkownika.

Począwszy od wersji 2.0,PC-DOS umożliwia użytkownikom instalowanie nowych urządzeń zewnętrznych w sposób standardowy na poziomie łącz DOS. W celu takiej instalacji programista powinien kolejno:

1. Napisać w asemblerze procedurę obsługi tego urządzenia.
2. Na dysku systemowym utworzyć plik o nazwie urządzenia i rozszerzeniu .COM, będący wynikiem asemblacji wyżej wymienionej procedury.
3. Jeżeli na dysku systemowym nie istnieje plik CONFIG.SYS – utworzyć go.
4. Do pliku CONFIG.SYS dołączyć polecenie: DEVICE = nazwa urządzenia .COM.

Każde urządzenie zewnętrzne ma przyporządkowany sobie nagłówek. Przy inicjacji systemu DOS instaluje nowe urządzenie wymienione w pliku CONFIG.SYS i włącza jego nagłówek w listę. Przy odwołaniu do jakiegokolwiek urządzenia DOS przeszukuje tę listę począwszy od ostatnio wprowadzonego nagłówka. Umożliwia to zastąpienie starych urządzeń zewnętrznych poprzez instalację nowych o tej samej nazwie.

POLECENIA

Polecenia PC-DOS, podobnie jak w systemach CP/M, dzielą się na rezydentne, tzn. stale obecne w pamięci operacyjnej i nierezydentne. Polecenia nierezydentne są zapisane na dysku systemowym w plikach typu .COM lub .EXE. Są one ładowane do pamięci w momencie odwołania się do nich.

Zestaw poleceń DOS jest stosunkowo rozbudowany. Rozszerzono znacznie, w stosunku do CP/M-86, ich liczbę, a równocześnie możliwości tego systemu. Pragnę zwrócić uwagę na niektóre zmiany:

● Wprowadzono polecenia operujące na drzewiastej strukturze katalogowej. Oto one:

.MKDIR (MaKe DIRectory) – utwórz katalog,
.RMDIR (ReMove DIRectory) – usuń katalog,
.CHDIR (CHange DIRectory) – zmień katalog aktualny,
.TREE – wyświetl drzewo katalogowe,
.PATH – wprowadź nową ścieżkę dostępu;

● Dodano polecenia umożliwiające sprawdzanie poprawności zapisywanych informacji. Oto one:

.VERIFY – sprawdza poprawność zapisu informacji na dysk poprzez ich odczyt i porównanie,

▶44

43 ◀

.COMP (COMPare) – porównuje zawartość 2 plików dyskowych (najczęściej polecenie to stosuje się bezpośrednio po poleceniu COPY),

.DISKCOMP (COMPare DISKettes) – porównuje zawartość 2 dyskietek (najczęściej polecenie to stosuje się bezpośrednio po poleceniu DISKCOPY);

● Wprowadzono mechanizmy ułatwiające odzyskanie plików z dysku z uszkodzonymi sektorami oraz dające możliwość wprowadzenia informacji zgubionych w wyniku tego uszkodzenia do pliku. Oto polecenia, które to umożliwiają:

.RECOVER – umożliwia przegranie plików z dysku, który ma uszkodzone sektory (oczywiście poza tymi sektorami),

.CHDSK (CHack DiSK) – analizuje katalogi i tablice alokacji plików na dysku oraz umożliwia wprowadzenie do pliku sektora, który został zniszczony;

● Dano możliwość podziału dysku stałego na 4 niezależne części. Umożliwia to używanie jednego dysku stałego przez kilka systemów operacyjnych. Wykorzystywane przy tym polecenia to:

.BACKUP – tworzy kopie plików z dysku stałego na dyskietkach,

.FDISK – operuje na partiach DOS,

.RESTORE – odtwarza pliki tymczasowo przechowywane na dyskietkach na dysku stałym.

● Dodano polecenia organizujące przetwarzanie wsadowe prac.

● Wprowadzono polecenia filtracyjne, które umożliwiają modyfikację danych przed ich wprowadzeniem na urządzenie wyjściowe. Oto one:

.SORT – sortuje dane tekstowe,

.FIND – przeszukuje tekst aż do znalezienia danego łańcucha tekstowego,

.MORE – organizuje wyświetlanie plików na konsoli.

● Dano możliwość wymiany procesora poleceń najwyższego poziomu COMMAND.COM (przy pomocy polecenia SHELL).

● Dodano polecenia zmieniające otoczenie systemu np.:

.PATH – wprowadza nową ścieżkę dostępu,

.PROMPT – ustala nowy znak zachęty.

● Umożliwiono zmianę standardowego wejścia lub wyjścia programu na dowolne urządzenie zewnętrzne lub plik dyskowy. Najlepiej wyjaśnią to przykłady:

.DIR>PRN – zawartość katalogu, zamiast na ekran, listowana jest na drukarce,

.DIR>DIRLIST – zawartość katalogu jest zapisywana do pliku DIRLIST,

.MYPROG<INPUT.TXT – plik INPUT.TXT staje się, zamiast klawiatury, wejściem do programu MYPROG,

.SORT<MYFILE>RESULT – plik MYFILE jest sortowany, a wynik zapisywany do pliku RESULT

(znak „<" oznacza zmianę standardowego wejścia do programu, „>" – wyjścia z programu.

● Dodano możliwość przetwarzania potokowego. Pozwala ono na natychmiastowe wykorzystanie pliku wynikowego jednego programu jako wejście do drugiego. Plik ten jest tworzony tymczasowo i w chwili gdy przestaje być potrzebny, jest usuwany zwalniając przydzielony mu obszar pamięci dyskowej, np.:

.DIR|SORT>FILE – zawartość katalogu jest sortowana, a następnie zapisywana do pliku FILE,

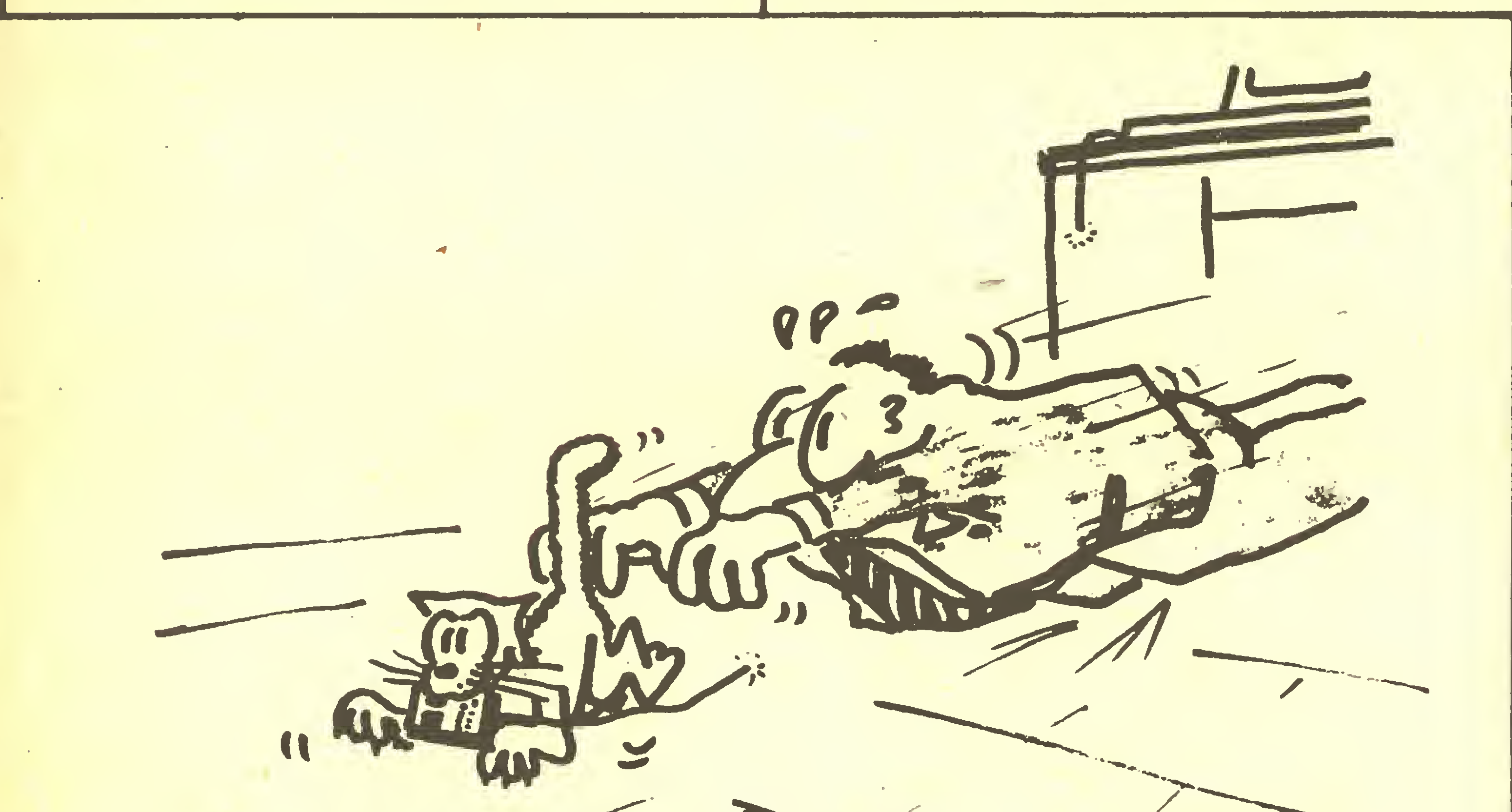
.PLM80|LINK|LOCATE>FILE – postać źródłowa programu w PL/M-80 jest przetwarzana na postać ładowalną, zapisaną w pliku FILE.

Niektóre z tych rozszerzeń (konkretnie 5 ostatnich) są zaczerpnięte z coraz bardziej popularnego systemu UNIX, a właściwie z jego wersji mikrokomputerowej, systemu XENIX.

ORGANIZACJA URUCHAMIANIA PROGRAMÓW UŻYTKOWYCH

Gdy program użytkowy jest wprowadzony do pamięci w celu wykonania go, DOS przydziela mu najniższy dostępny obszar pamięci, zwany do tej pory segmentem programu. W pierwszych 256 bajtach tego obszaru DOS buduje tzw. wstępny blok sterujący, który jest odpowiednikiem strony zerowej w systemach CP/M. Program użytkowy jest umieszczany powyżej tego obszaru, począwszy od adresu 100H (adres względny).

Programy użytkowe mogą korzystać z bogatego repertuaru funkcji systemowych DOS. Organizują one transmisje z zewnętrznymi urządzeniami znakowymi, sterują plikami, zarządzają pamięcią, ustalają czas i datę, umożliwiają wykonanie innych programów. Są odpowiednikiem funkcji BDOS w CP/M. Podobnie jak w systemach CP/M, w PC-DOS można korzystać w sposób bezpośredni z procedur obsługi urządzeń zewnętrznych zapisanych w BIOS-ie, ale zalecane jest organizowanie transmisji na wyższym poziomie, za pośrednictwem funkcji systemowych.



OCENA SYSTEMU

PC-DOS jest nowoczesnym systemem operacyjnym. Jego twórca wzorował się na nieco zmodyfikowanej wersji systemu CP/M (związki między tymi systemami są oczywiste) oraz analizował wady i zalety różnych systemów nie starając się o zachowanie kompatybilności z którymkolwiek z nich. Po prawie roku używania DOS powstała jego druga wersja, częściowo kompatybilna z poprzednią, za to wprowadzająca wiele cech systemu UNIX. Została ona entuzjastycznie przyjęta przez użytkowników.

Pragnę zwrócić uwagę na niektóre z zalet DOS. Oto one:

● posiada krótką i prostą składnię poleceń, nie używa zbędnych znaków interpunkcyjnych;

● ma hierarchiczną drzewiastą strukturę plików, skracającą czas dostępu do nich i pozwalającą na ich grupowanie;

● umożliwia podział dysku stałego na części, na każdej z nich może pracować inny system operacyjny;

● można korzystać z poleceń nie zapisanych w katalogu aktualnym po podaniu ścieżki dostępu poleceniem PATH;

● posiada najbardziej popularne cechy systemu UNIX: filtry i przetwarzanie potokowe;

● daje możliwość tworzenia pliku komend zwanego plikiem wsadowym, który jest wykonywany bez interwencji użytkownika i steruje przetwarzaniem grupowym prac;

● udostępnia łatwy, standardowy sposób dołączania urządzeń zewnętrznych;

● umożliwia drukowanie plików, podczas gdy system pracuje nad innymi zadaniami;

● ma możliwość odtwarzania i poprawiania częściowo zniszczonych plików;

● może sprawdzać poprawność zapisu informacji na dysk;

● jest elastyczny, dający się łatwo przystosować do upodobań i przyzwyczajeń użytkowników.

PC-DOS nie jest jednak pozbawiony wad. Pragnę zwrócić uwagę na następujące:

● brak jakiejkolwiek ochrony plików;

● co prawda DOS może korzystać z nieobecnych w aktualnym katalogu poleceń, których ścieżka dostępu podana jest poleceniem PATH, ale programy te nie mogą mieć żadnych nakładek. Są bowiem poszukiwane jedynie w katalogu bieżącym.

PC-DOS bezsprzecznie został uznany za standard wśród 16-bitowych systemów operacyjnych i jak na razie nic nie wskazuje na to, aby miał utracić tę pozycję.

DANUTA MAGDZIK

PĘTLICZEK – bo pętla jest podstawą programowania. Tu znajdziesz kolejną porcję zadań naszego Klubu Mistrzów Komputera*.
MĘTLICZEK – bo znajdziesz tu różne różności, związane z minikomputerem tak cienką nitką, że Redakcja już nie bierze za nią odpowiedzialności.

* regulamin KMK w numerze 2'86 naszego pisma.

SŁOWO DO ZADAŃ

W trzech poprzednich numerach opublikowaliśmy wszystkie 9 zadań z pierwszej serii zadań klubowych. Na rozwiązanie czekamy do końca października. Biorąc pod uwagę cykl produkcyjny naszego miesięcznika, wyniki podamy w styczniu przyszłego roku.

Dziś rozpoczynamy serię drugą. Przypominam, że aby zostać członkiem naszego Klubu, należy z danej serii 9 zadań przysłać w terminie 3 miesięcy rozwiązania co najmniej sześciu z nich. Ponadto w tym samym terminie należy nadesłać co najmniej dwa oryginalne, ułożone przez siebie zadania. Członkowie Klubu, a w tej chwili ci, którzy zostaną członkami Klubu po pierwszej serii zadań, muszą spełnić połowę wymagań, a więc rozwiązać trzy zadania i ułożyć jedno.

Zadania są zwykle formułowane w postaci: napisać program, który... . Przyjmujemy programy napisane w dowolnym, powszechnie znanym języku programowania, dostępnym na popularnym mikrokomputerze. Najwygodniej dla redakcji byłoby, gdyby programy te docierały do nas w formie nagrania na kasecie.

Klub nasz jest dostępny również dla tych, którzy nie posiadają komputerów. Jako rozwiązania przyjmowane będą także opisy alogrytmów lub schematy rozwiązań postawionych problemów.

Jeszcze jedna uwaga. Ponieważ najpopularniejszym mikrokomputerem w Polsce jest ZX Spectrum, to zadania ściśle związane z systemem formułowane są jakby na ten komputer. Oczywiście przyjmujemy rozwiązania tych zadań, odpowiednio przeformułowanych na inne maszyny.

SZYBCIEJ NIŻ KOMPUTER

Czy potraficie obliczyć w pamięci wyniki następujących wyrażeń szybciej niż trwałoby wpisywanie ich do komputera?

(a) $12345 + 54321 = ?$

(b) $\dfrac{8642 + 1357}{2468 - 1357} = ?$

(c) $\dfrac{543210}{54321^2 - 543210 \times 543212} = ?$

Zadanie (c) pochodzi ze zbioru "Rozkosze łamania głowy" L. Pijanowskiego.

Z tej samej książki pochodzi następujące zadanie: "Mam trzy jednakowe cegły o wymiarach 45×22.5×15 centymetrów. Mogę układać jedną cegłę na drugiej w dowolny sposób.

Ile piramid o różnych wysokościach uda mi się zbudować z trzech cegieł?"

Spróbujcie rozwiązać to zadanie za pomocą komputera. Może komuś uda się zmusić komputer, by narysował wszystkie piramidy.

HISTORIA KOMPUTERA

2. Kamienie jako ROM

Szybko okazało się, że ręka jako maszyna licząca nie spełnia wszystkich oczekiwań. Głównym powodem niezadowolenia była niestabilność przechowywanych informacji. Jeżeli w trakcie liczenia pojawił się w pobliżu choćby komar, to z dużym prawdopodobieństwem następowało przekłamanie, ręka bowiem służyła do wielu celów. W takiej sytuacji trzeba było zaczynać obliczenia od początku. Pojawiła się konieczność wyposażenia komputera w pamięć stałą.

Pierwszym dobrym pomysłem było użycie drobnych kamieni. Zliczając przedmioty nasz prapradziad odkładał w oznaczone miejsce kamienie. Potem mógł je spokojnie przeliczyć przy pomocy znanego już wcześniej urządzenia: ręki. Takie rozwiązanie miało wielką zaletę. Można już było wykonywać dodawanie i odejmowanie. Dodawanie, na przykład, wykonywano przez zsypanie dwóch stert kamieni i liczenie ich razem. Wadą takiego systemu było natomiast to, że liczący nigdy nie miał pewności czy dany kamień leżał w tym miejscu wcześniej, czy też został dołożony w trakcie obliczeń. Stanowiło to duży mankament, zwłaszcza w terenach kamienistych.

(cdn.)



PRAWO MURPHY'EGO:

2. Jeżeli coś może popsuć się w wielu miejscach, to pierwsze uszkodzenie wystąpi tam, gdzie wyrządzi najwięcej szkód.

ZADANIA KLUBOWE – SERIA II

Bardzo prostą, a jednocześnie ciekawą grą dla dwóch osób, jest gra w KOSTKĘ. Jedynym rekwizytem w tej grze jest zwykła, prawidłowa kostka do gry. Pierwszy gracz rzuca kostką i odczytuje liczbę oczek na górnej ściance. Będzie to liczba początkowa. Od tego momentu gra jest ściśle strategiczna. W kolejnym ruchu gracz przewraca kostkę o 90°, tzn. kładzie ją na jednej z pionowych ścianek i do liczby poprzednio uzyskanej dodaje liczbę oczek na "nowej" górnej ściance. Ruchy wykonywane są na zmianę, aż do uzyskania pewnej, z góry określonej liczby. Kto pierwszy uzyska lub przekroczy tę liczbę, ten wygrywa.

Proponuję napisać program zastępujący jednego z graczy. Zamiast programu można przysłać opis strategii w tej grze. Mile widziane będą programy realizujące odmiany tej gry.

* * *

Liczbami trójkątnymi nazywamy liczby postaci $\dfrac{n \times (n+1)}{2}$ gdzie n jest liczbą całkowitą większą od zera. Liczba taka wyraża sumę wszystkich dodatnich liczb całkowitych nie większych od n. Nazwa pochodzi stąd, że jeżeli liczba przedmiotów jest liczbą trójkątną, to możemy ułożyć te przedmioty w trójkąt:

$$\frac{3 \times 4}{2} = 6$$

O liczbach trójkątnych można udowodnić np. następujące twierdzenie: "istnieją liczby trójkątne o dowolnej liczbie cyfr".

Proponuję napisać program lub opisać algorytm znajdujący dla danej liczby k liczbę trójkątną mającą dokładnie k cyfr (nieistotne zera "z przodu" liczby pomijamy).

* * *

Język programowania Basic uprawnia swoją budową do tworzenia bałaganu w programie. Przyczyną jest nadużywanie instrukcji skoku, bez której jednak nie można się obyć.

Proponuję napisać program analizujący źródłowy program w Basicu i wypisujący tzw. mapę odwołań. To znaczy mamy otrzymać informacje, gdzie zaczynają i kończą się pętle (nie tylko FOR/NEXT), gdzie używane są instrukcje skoku i dokąd prowadzą, gdzie mamy podprogramy itd.

Komputer i ∞

Szanowny Panie Redaktorze!

Czy zastanawiał się Pan kiedyś nad sposobem, w jaki budujemy zdania, gdy coś mówimy lub piszemy? Z pewnością wielokrotnie jest Pan zmuszony poprawiać zdania, które są niejasne, przestawiać szyk wyrazów, by zdanie lepiej brzmiało. Wykonuje Pan tę pracę, jak mi się wydaje, myśląc głównie o tym, żeby adiustowany tekst był łatwiejszy do przeczytania i zrozumienia. Chcę jednak zwrócić Pana uwagę na teoretyczną budowę zdania.

Wszyscy uczyliśmy się w szkole, że poprawnie zbudowane zdanie może mieć w języku polskim na przykład taki schemat:

PRZYDAWKA PODMIOT ORZECZENIE DOPEŁNIENIE

Aby z tego ciągu słów otrzymać zdanie, musimy wykonać kilka operacji. Operacje te to zastępowanie nazw części zdania przez słowa mogące pełnić w zdaniu odpowiednią funkcję. Ponieważ podmiotem może być rzeczownik, więc słowo *PODMIOT* możemy zastąpić słowem "komputer", otrzymując ciąg:

PRZYDAWKA komputer ORZECZENIE DOPEŁNIENIE.

Ciąg ten nadal nie ma sensu jako zdanie języka polskiego, ale ma potencjalne możliwości stać się poprawnym zdaniem. By tego dokonać, musimy zastąpić pozostałe nazwy części zdania odpowiednimi słowami. Przy tej zamianie trzeba pamiętać, że nie mamy tu pełnej dowolności. Nazwę *ORZECZENIE* można zastąpić jedynie czasownikiem (proszę mi wybaczyć uproszczenia), a nazwę *PRZYDAWKA* – przymiotnikiem. Nie są to jednak jedyne reguły. To, czy zdanie stanie się poprawne po wykonaniu wszystkich zamian, zależy też od form gramatycznych. Na przykład decyzja o zastąpieniu nazwy *PODMIOT* wyrazem *komputer* narzuca już pewien kontekst. Teraz nie możemy w miejsce *PRZYDAWKA* wstawić słowa *zielona*, mimo że jest ono przymiotnikiem. Podobnie w miejsce *ORZECZENIE* możemy wstawić tylko czasownik w trzeciej osobie liczby pojedynczej. Respektując te zasady otrzymamy w końcu zdanie

Dobry komputer oblicza całki.

Nie jest tutaj ważna treść tego zdania, lecz jego poprawność gramatyczna.

Gdy tak patrzymy na gramatykę języka naturalnego, to dzieli nas tylko krok od stworzenia modelu matematycznego gramatyki. Krok taki został dokonany w latach pięćdziesiątych naszego stulecia przez matematyków zajmujących się badaniem języków naturalnych, czyli lingwistyką matematyczną. Wprowadzone pojęcie otrzymało nazwę gramatyki generującej. Obiekt taki składa się z dwóch słowników. Pierwszy z nich zawiera symbole pomocnicze – dla nas to będą nazwy części zdania. Elementy drugiego słownika to symbole końcowe, a więc w naszym przypadku słowa języka polskiego. Gramatyka jest jeszcze wyposażona w zbiór produkcji. Produkcje to reguły zamiany jednego ciągu słów na drugi. Zamiast podawać formalną definicję gramatyki generującej omówię prosty przykład.

Rozpatrzmy gramatykę zawierającą jeden symbol pomocniczy *A* oraz dwa symbole końcowe *a, b*. Niech jedynymi produkcjami będą: *A — aAb, aA – – – b*. W przypadku pierwszej produkcji jeżeli w budowanym zdaniu występuje symbol *A*, to możemy go zamienić na ciąg *aAb*, zostawiając resztę zdania bez zmian. Druga produkcja (będąca produkcją kontekstową) oznacza, że jeżeli w zdaniu występuje symbol *A* po prawej stronie symbolu *a*, to oba symbole można zastąpić jednym symbolem *b*.

Popatrzmy teraz jak tworzy się zdanie w języku z taką gramatyką. Zaczynamy od schematu zdania złożonego z jednego symbolu pomocniczego *A* (w definicji gramatyki zawsze wyróżnia się jeden symbol pomocniczy jako symbol początkowy). Wykonując dwa razy pierwszą produkcję, a następnie drugą, otrzymamy

A – – – aAb – – – aaAbb – – – abbb.

Tak więc zdanie abbb jest gramatycznie poprawnym zdaniem w naszym języku. Zapewne analogie między strukturą taką jak w tym przykładzie a sposobem budowy zdania w języku polskim są dla Pana bardzo wyraźne.

Spodziewam się, że zauważył Pan pewne podobieństwo między gramatykami a automatami, o których pisałem poprzednio. Rzeczywiście, definicje są podobne, różne są natomiast role obu klas obiektów. Automaty to głównie akceptory – urządzenia sprawdzające, czy dany ciąg jest poprawny, czy nie. Gramatyki natomiast służą do tworzenia poprawnie zbudowanych zdań w języku, który opisują.

Chyba już wyczerpałem Pańską cierpliwość na rozważania lingwistyczno-matematyczne, czas więc wrócić do komputera. Zacznę od problemu, który był i jest jednym z motorów lingwistyki matematycznej. Jest to problem automatycznego przekładu tekstu z jednego języka naturalnego na drugi. Taki przekład oczywiście nie może odbywać się przez zwykłą zamianę słów na przykład angielskich na polskie bez zmiany ich miejsca w zdaniu. Zasady gramatyki są niezbędne do tworzenia zrozumiałego tekstu w języku, na który się tłumaczy oraz do zrozumienia tekstu tłumaczonego. Gdyby dało się opisać język naturalny za pomocą gramatyki generującej, to znaczy dobrać odpowiednie zbiory symboli pomocniczych i końcowych oraz zadać produkcje, tak by za pomocą tej gramatyki można było utworzyć znaczącą liczbę poprawnych zdań w języku naturalnym, to byłaby to połowa sukcesu. Można sobie bowiem wyobrazić, że mając dwie gramatyki będziemy w stanie określić reguły mówiące, jakim symbolom i produkcjom jednej gramatyki odpowiadają symbole i produkcje drugiej. Uzyskalibyśmy dzięki temu możliwość tłumaczenia poprawnego zdania jednego języka na poprawne zdanie drugiego języka. To byłoby już coś. Niestety, jak się Pan zapewne domyśla, konstrukcja gramatyki generującej dla języka naturalnego to sprawa trudna i skomplikowana – tak jak skomplikowane są języki używane w życiu codziennym.

Trochę inaczej przedstawia się sprawa z językami sztucznymi. Doskonale Pan zna, jak przypuszczam, kilka takich języków. Mam na myśli oczywiście języki programowania. Do badania tych języków gramatyki generujące doskonale się nadają. Znane są również translatory, czyli programy tłumaczące programy napisane w jednym języku na równoważne im programy w innym języku.

Teoria gramatyk formalnych jest już samodzielną dyscypliną. Bada się gramatyki o różnych własnościach, języki tworzone przez te gramatyki, złożoność wyprowadzania poprawnych zdań oraz wiele innych problemów. Dla matematycznych podstaw informatyki ważną klasą były gramatyki bezkontekstowe, to znaczy takie, w których produkcje podają tylko, jakim ciągiem można zastąpić symbol pomocniczy, bez względu na kontekst (otoczenie), w którym on występuje. Gramatyki bezkontekstowe wprowadził N. Chomsky w 1956 r. badając języki naturalne. Popularność tych gramatyk wzrosła, gdy w 1962 roku udowodniono, że języki generowane przez te gramatyki są tzw. językami Algolo-podobnymi. Języki Algolo-podobne zaś stanowiły dla matematyków intuicyjny model języka programowania. Niestety, wkrótce okazało się, że języki Algolo-podobne są zbyt słabe, by być dobrym modelem dla języków programowania. W szczególności sam Algol nie jest językiem Algolo-podobnym, a więc nie jest generowany przez gramatykę bezkontekstową. Obecnie większe zainteresowanie budzą języki kontekstowe, generowane przez gramatyki dopuszczające kontekstowe produkcje.

Myślę, że będzie jeszcze okazja powrócić do teorii języków formalnych i teorii gramatyk choćby pokazując, jak można definiować języki programowania.

Załączam pozdrowienia dla Czytelników
Pańskiego pisma
MATEMATYK

Ps. Chciałbym jeszcze przedstawić jedno twierdzenie dotyczące języków definiowanych przez gramatyki generujące. Otóż udowodniono, że klasa wszystkich języków definiowanych gramatykami generującymi jest identyczna z klasą wszystkich języków rekurencyjnie przeliczalnych. Znaczy to, że dla każdego języka, dla którego istnieje gramatyka definiująca go, istnieje też funkcja rekurencyjna, przy której obrazem zbioru liczb naturalnych jest ten właśnie język.

input ⇔ output

Szanowna Redakcjo!

Od dwóch miesięcy jestem użytkownikiem komputera Atari 800XL. Obcojęzyczna instrukcja języka Atari Basic jest bardzo ogólnikowa i trudna do przetłumaczenia dla początkującego "informatyka". Dlatego też jestem zmuszony do korzystania przy programowaniu z ogólnodostępnej literatury (głównie czasopism). Jestem nauczycielem fizyki w szkole podstawowej w niewielkim mieście i nie mam kontaktu z żadnym klubem komputerowym. W większości programów, które próbuję napisać, napotykam na bariery wynikające z różnic pomiędzy Atari Basic a innymi wersjami języka Basic.

W imieniu użytkowników Atari proszę o opis języka Atari Basic i obsługi tego komputera ze szczególnym uwzględnieniem jego rozbudowanej grafiki.

Proszę także w miarę możliwości o odpowiedź na następujące pytania:

1. W jaki sposób zmierzyć czas oczekiwania komputera na odpowiedź (licznik czasu)?

2. W jaki sposób programować ruchome elementy obrazu?

3. W jaki sposób zrealizować na Atari instrukcje PLOT INVERSE 1, PLOT OVER 1 i POINT, dostępne na ZX Spectrum.

Janusz Berładyn
Barlinek

Wśród napływających do nas listów te od początkujących użytkowników Atari 800XL stanowią sporą grupę. We wszystkich znajdujemy prośby o publikacje szczegółowych informacji dotyczących obsługi komputera, listingów programów użytkowych czy też gier. Oczywiście będziemy starali się spełniać te prośby. Dziwi nas tylko fakt, że wiele listów dotyczy także publikacji instrukcji obsługi. Czyżby importowany przez firmę PEWEX komputer był sprzedawany bez niej?

Jako eksperci od spraw Atari będą występowali członkowie znanego w kraju krakowskiego Klubu Użytkowników Atari, których zaprosiliśmy do współpracy. W bieżącym numerze na pytania Czytelnika odpowiada Mariusz Giergiel. (red)

1. Czas oczekiwania komputera na odpowiedź

W systemie operacyjnym mikrokomputerów Atari ustalony jest dla poszczególnych urządzeń tzw. czas oczekiwania na odpowiedź. Jeżeli po upływie tego czasu urządzenie nie zgłasza się, tzn. nie jest "widziane" przez komputer, występuje błąd nr 138. Dla przykładu dla magnetofonu czas oczekiwania na odpowiedź wynosi ok. 20 sekund.

Jeżeli w czasie użytkowania komputera wystąpi błąd nr 138, należy sprawdzić prawidłowość połączeń między urządzeniem a komputerem. Sprawdzić czy do urządzenia podłączone jest napięcie zasilania, czy urządzenie przełączone jest na tryb "on line" (częste przy drukarkach).

W przypadku gdy wszystkie połączenia są prawidłowe błąd nr 138, może świadczyć o uszkodzeniu komputera lub urządzenia peryferyjnego.

Do zmierzenia czasu można wykorzystać zawarty w systemie operacyjnym zegar czasu rzeczywistego. Komórki systemowe 18, 19 i 20 zawierają licznik ramek, zwiększający swój stan o 150 razy na sekundę (w wersji PAL komputera). W oparciu o nie można odmierzać czas np. przy pomocy wyrażenia:

czas=INT((65536×PEEK (18) +256×PEEK (19) +PEEK (20)) /50).

Obliczony w ten sposób czas będzie odmierzany w sekundach. (mg)

2. Atari player/missile graphics.

Możliwości animacji obrazu to jedna z ciekawszych cech komputerów domowych. Animację można wykonywać różnymi sposobami, najlepsze efekty jednak uzyskuje się stosując właśnie grafikę obiektów P/M, zwanych też niekiedy SPRITE'ami. Istotą grafiki P/M jest sposób pamiętania ruchomego obiektu poruszającego się po dwuwymiarowym ekranie, dostosowany do liniowej struktury (jednowymiarowej) pamięci RAM ekranu a także specjalne rozwiązania sprzętowe.

Atari umożliwia zdefiniowanie czterech obiektów typu P od P0 do P3, które poruszają się niezależnie i których kolory (określone w rejestrach koloru obiektów P/M) są także niezależne od siebie i od pozostałych kolorów na ekranie. Każdy z obiektów od P0 do P3 może być przedstawiany: w wielkości normalnej, w podwójnej szerokości albo też w podwójnej wielkości tzn. szerokości i wysokości (analogicznie jak litery w trybach graficznych 0, 1 i 2). Z każdym z obiektów typu P może być stowarzyszony obiekt typu M, odpowiednio od M0 do M3. Kolory obiektów typu M są takie jak obiektów typu P, którym odpowiadają, natomiast sposób poruszania się jest niezależny od obiektów P jak i od obiektów M. Z czterech obiektów M można utworzyć jeden dodatkowy, piąty obiekt P4 (jego kolor jest wtedy taki, jak obiektu P3). W normalnych rozmiarach obiekt P może mieć osiem punktów wysokości, natomiast obiekt M – dwa punkty.

Poruszaniem obiektów P/M i testowaniem ich kolizji zajmuje się specjalizowany układ GTIA. Informacje o ewentualnych kolizjach mogą być wykorzystywane przez program użytkownika.

Możliwości wykorzystania grafiki P/M są duże i różnorodne, mogą dotyczyć nie tylko animacji, ale i np. definiowania własnych znaków graficznych nie mieszczących się w siatce ekranu, takich jak indeksy we wzorach matematycznych itp. Montowany fabrycznie interpreter języka Atari Basic nie ma instrukcji umożliwiających tworzenie i animację obiektów P/M oraz wykorzystanie grafiki P/M. Opisanie podstaw tworzenia i animacji P/M w Atari Basic wymagałoby oddzielnego, obszernego artykułu. Wygodne instrukcje do tworzenia i animacji P/M posiada natomiast rewelacyjny Basic XL dla Atari.

Zainteresowanym wykorzystaniem grafiki P/M polecić można książkę "Your ATARI Computer" wydaną przez OSBORNE/McGraw–Hill w 1982 roku. (mg)

3. PLOT, COLOR i LOCATE

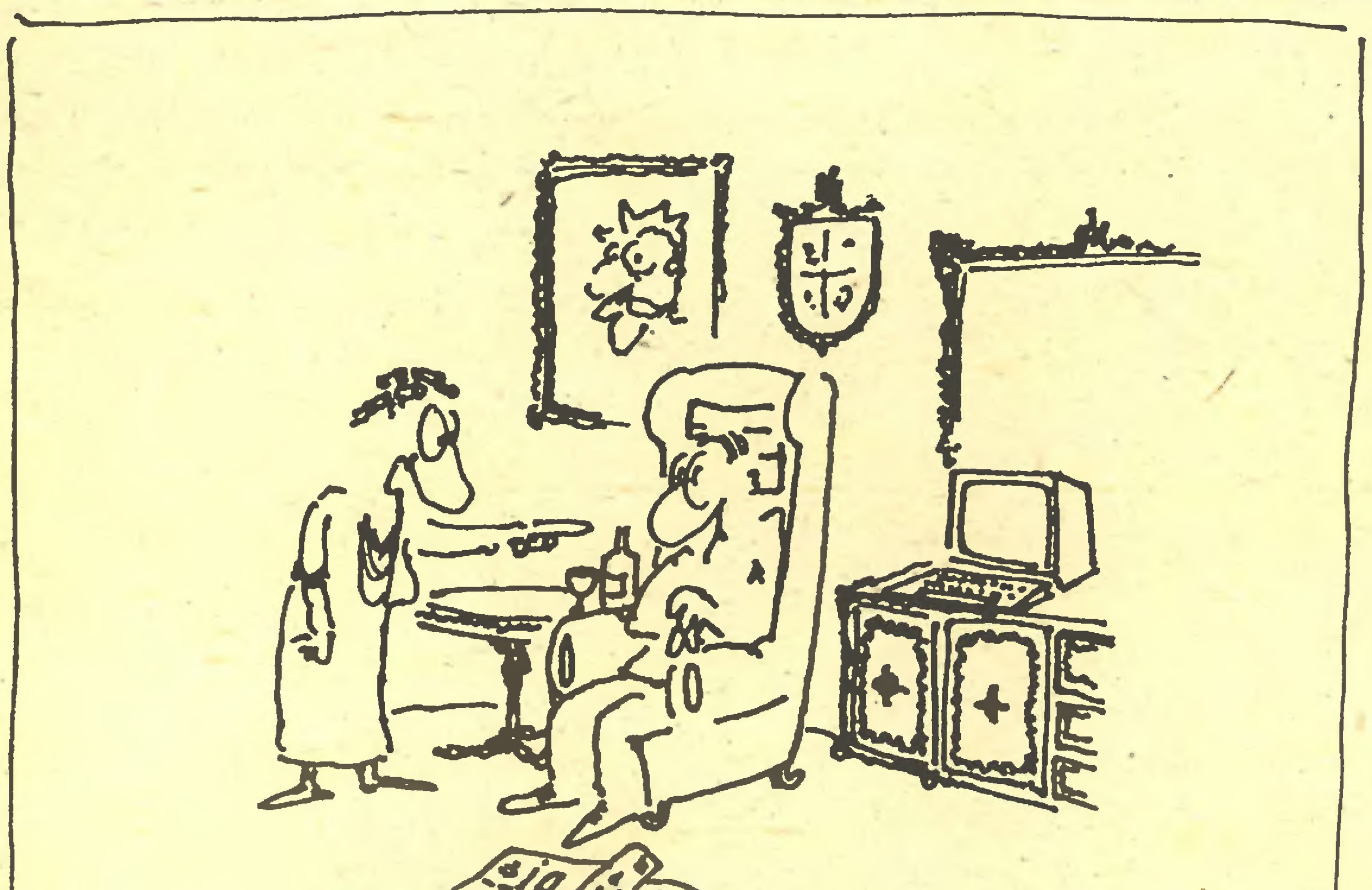
Dostępne w języku Basic na mikrokomputerze ZX Spectrum wymienione instrukcje można zrealizować w następujący sposób:

● PLOT INVERSE 1; x, y powoduje wygaszenie zapalonego punktu, natomiast w przypadku gdy punkt nie był zapalony, nie dzieje się nic. Podobny do tego skutek na mikrokomputerze Atari można otrzymać stosując instrukcje COLOR n i PLOT x, y. W instrukcji COLOR n, n oznacza numer rejestru z kolorem tła (wynosi on z reguły 4, dla trybów graficznych od 3 do 10, za wyjątkiem trybu 8, gdzie n=2). Oczywiście po zgaszeniu punktu należy ponownie instrukcją COLOR n przywrócić poprzednią wartość rejestru;

● PLOT OVER 1; x, y na zasadzie flip–flop gasi punkt zapalony i zapala punkt, gdy ten się poprzednio nie palił. O ile w przypadku gaszenia punktu sprawa jest w miarę prosta (patrz powyżej), to dla przypadku zapalania punktu sytuacja jest nieco bardziej skomplikowana, gdy należy rozważyć, w jakim kolorze powinien być on zapalany. Reasumując, zastąpienie tej instrukcji wymaga oddzielnego podprogramu.

● POINT x, y to funkcja testująca stan pixela (punktu na ekranie). Gdy jest on zapalony, jej wartość wynosi 1, a gdy zgaszony 0. Odpowiednikiem powyższej funkcji w Atari jest LOCATE x, y, n, gdzie jako n zwracany jest numer rejestru koloru, w którym podany jest kod koloru punktu w trybie graficznym. W przypadku jeżeli punkt był zgaszony, podawany jest numer rejestru koloru tła.

(red.)



– WIDZISZ SINCLAIR, ZAWSZE CI MÓWIŁAM ŻE TO DIABELSKIE URZĄDZENIE DO PROWADZI CIĘ DO UPADKU!

Giełda

WARSZAWA

Giełdę sprzętu mikrokomputerowego w studenckim klubie Stodoła miłośnicy komputerów znają doskonale. Obecnie giełdzie tej wyrosły konkurentki. Otóż w soboty można wybrać się na giełdę organizowaną w Szkole Podstawowej Nr 25 w Warszawie przy ulicy Grzybowskiej. Ceny i sprzęt podobny jak na innych giełdach.

● Commodore C 128	300 tys. zł
C 64	160 tys. zł
● Stacja dysków 1541	150 tys. zł
● Stacja dysków 1570	270 tys. zł
● Atari 800 XL	95 tys. zł
● Atari 130 XE	150 tys. zł
● Amstrad 464 + zielony monitor	220 tys. zł
● Dyskietki 3,5 cala	3000 zł
● Dyskietki 5,25 cala	1200 zł
● Interfejs 1 + microdrive do ZX Spectrum	90 tys. zł

Poza bogatym oprogramowaniem do ZX Spectrum pojawia się coraz więcej programów i literatury do Atari, ta ostatnia często w tłumaczeniach.

* * *

Najmłodszą giełdą mikrokomputerów jest giełda w klubie studenckim Hybrydy. Odbywa się ona w drugą i czwartą niedzielę miesiąca. Impreza jest jeszcze mało znana i w związku z tym na giełdzie więcej oferujących niż oglądających, a kupujących brak. Jeżeli dopiszą w następne niedziele uczestnicy, to w Hybrydach powtórzy się sytuacja ze Stodoły (ciemno, ciasno, duszno).

Wprowadzony w maju zakaz sprzedaży części i sprzętu elektronicznego na warszawskim "perskim jarmarku" poważnie utrudni majsterkowiczom zdobycie nieobecnych w handlu detalicznym elementów i części systemów mikrokomputerowych. Poniżej kilka cen z okresu, kiedy wolno było handlować:

● Pamięci EPROM 2716	1700 zł
2764	2900 zł
27128	4500 zł
● ROM do ZX Spectrum	6000 zł
● Procesory 8085	2700 zł
Z80	3000 zł
● Układy serii 74 LS...	350-1200 zł
● Napęd dyskowy 3 cale (płytka elektroniki + napęd dysku i głowica)	100 tys. zł
● Złącze krawędziowe 96-stykowe	1300 zł

(ZR)

* * *

Przed miesiącem informowaliśmy o rozpoczęciu sprzedaży komputerów po wysokich – jak nam się zdawało – cenach we wzorcowym sklepie spożywczym WSS "Społem" "ABC" w Warszawie. Jak poinformował nas kierownik stoiska w ciągu pierwszego miesiąca obroty przekroczyły 100 mln zł. Dostawcą sprzętu jest BOMIS.

Gratulujemy – i powtarzamy swą opinię, że zjawisko to świadczy, że nasz rynek komputerowy dojrzewa do powstania normalnej sieci handlowej, w której dokonuje się zakupu bez zamówień i kolejek.

(WM)

KRAKÓW

W Krakowie ceny są ustabilizowane. Oto notowania z giełdy w klubie Karlik 15 czerwca:

● ZX 81	30 tys. zł
● ZX Spectrum 48 K	80-85 tys. zł
● ZX Spectrum Plus	110-120 tys. zł
● Atari 800 XL	75-80 tys. zł (ceny spadły, co chyba wiąże się z obniżką w PEWEX-ie)
● Atari 800XL + joystick + magnetofon	125 tys. zł (duża podaż)
● Atari 130XE	155-160 tys. zł
● Sharp MZ 721	130-170 tys. zł
● Sharp MZ 731	180 tys. zł
● Commodore 116	38-45 tys. zł
● Schneider CPC464 + zielony monitor	250 tys. zł
● Moduł PAL/SECAM do Jowisza	23 tys. zł
● Procesor 8080	3 tys. zł
● Instrukcja Spectrum Basic w języku polskim (Polbrit)	2 tys. zł
● Dyskietki 5,25 cala	1200-1300 zł (duża podaż)

Ceny w komisach były oczywiście wyższe:

● ZX Spectrum 48K	130 tys. zł
● ZX 81 + 16K pamięci	40 tys. zł
● Atari 800XL	120 tys. zł
● Atari 800XL + magnetofon	150 tys. zł
● Atari 130XE	200 tys. zł
● Commodore 116	60 tys. zł
● Commodore 16	100 tys. zł
● Dyskietki (za 10 sztuk)	18-20 tys. zł

notował ANDRZEJ ZAŁUSKI

KASETY KOMPUTERA

Czy kupiłeś już kasety z programami "Komputera"?

Jeśli nie – możesz jeszcze nadrobić swoją opieszałość w jednym z Klubów Międzynarodowej Prasy i Książki:

1. Bydgoszcz	– Al. 1 Maja 10
2. Gdańsk	– Długi Targ 25/27
3. Katowice	– ul. Wawelska 2
4. Gliwice	– Rynek Główny
5. Kielce	– Rewolucji Październikowej 3/5
6. Koszalin	– Zwycięstwa 106/108
7. Kraków	– Mały Rynek 4
8. Lublin	– Krakowskie Przedmieście 20
9. Łódź	– ul. Narutowicza 8/10
10. Olsztyn	– ul. 1 Maja 18/19
11. Rzeszów	– ul. Słowackiego 11
12. Poznań	– ul. Ratajczaka 39
13. Szczecin	– Al. Wojska Polskiego 2
14. Warszawa	– ul. Bagatela 14
15. Warszawa	– ul. Marszałkowska 116/122
16. Wrocław	– Pl. Kościuszki 21/23

Doskonałe programy. Opisy. Rewelacyjne ceny.